ARGO-VLA: CALIBRATED GATING AND ESCALATION FOR CORRECTING AND ORCHESTRATING FROZEN VLA POLICIES IN THE LABORATORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) policies fine-tuned for laboratory skills fail at two timescales. Within a skill they miss millimeter tolerances: on AutoBio (Lan et al., 2026), a fine-tuned π_0 succeeds on at most 43% of episodes of any medium task and 15% of any success-scored hard task. Across a protocol they cannot order steps, withhold hazardous operations, or notice that a step failed. We propose ARGO-VLA, built around one calibrated novelty score used at both timescales. At the control rate the score masks a bounded residual, trained by behavior cloning and PPO on a chunk-level residual Markov decision process and restricted to its training support by a k-nearest-neighbor trust gate, that corrects each action chunk of a frozen π_0. At the protocol-step rate the same score, thresholded by split conformal prediction, halts a skill and escalates to a typed state machine of LLM agents that plans, validates and risk-tags, requests human approval, executes and reviews protocol steps; the threshold bounds the false-escalation rate per nominal skill call. We introduce AnomalyBio, which extends AutoBio with multi-step protocols, injected failures and hazardous or compositionally infeasible goal probes judged by simulator ground truth, and ablate the trust gate, anomaly mask, escalation, validator critique and LLM reviewer. On *transfer*, the residual reaches 48.7% success versus 40.7% for AutoBio's published π_0, a comparison across different checkpoints and episodes, not a controlled one. Escalation detects 52% of monitor-reachable injected failures at 1.8% false escalations per call, and moves protocol success under injected failures from 45.3% to 48.0%, a gain within one standard error.

1 INTRODUCTION

Laboratory protocols are procedural, individual steps are contact-rich and tolerance-critical (seating a tube in a rotor slot, placing a pipette tip below a liquid surface), and some errors are irreversible (overheating a sample, discarding the wrong tube). VLA models such as π_0 (Black et al., 2025b) can be fine-tuned on such skills, and AutoBio (Lan et al., 2026) measures them in a simulated biology laboratory. With 100 demonstrations, π_0 reaches 40.7% on *transfer centrifuge tube*, 42.7% on *aspirate with pipette* and 14.7% on *load centrifuge rotor*, and the benchmark's authors attribute failures on contact-rich tasks to open-loop execution without closed-loop recovery (Lan et al., 2026).

Making such a robot reliable faces four challenges. First, many failures inside a skill are small, systematic offsets in otherwise correct action chunks. Full-model RL fine-tuning (Li et al., 2026; Chen et al., 2025) can reduce them but is expensive for a 3.3B-parameter flow model, and a lightweight learned correction is trustworthy only on states like those it was trained on. Second, a dropped tube, displaced labware or an occluded camera puts the robot where any learned correction extrapolates and the skill should stop; detection must run at the control rate, and its false alarms must be bounded, since each one interrupts a protocol. Third, across steps the robot must order skills, withhold hazardous operations until a human approves them, and recover from a failed step without repeating irreversible ones. LLM planners decompose goals but verify their own plans poorly (Kambhampati et al., 2024; Huang et al., 2024), and hierarchical VLM-over-VLA and dual-system models (Shi

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.

et al., 2025; Black et al., 2025a; NVIDIA et al., 2025) target instruction following rather than correction or calibrated failure detection. Fourth, no existing benchmark measures failure detection and recovery in laboratory protocols against simulator ground truth.

We introduce **ARGO-VLA**, a framework in which one calibrated novelty score couples correction inside a skill with orchestration across a protocol. It has three tightly coupled components. The **trust-gated chunk residual** corrects each action chunk of a frozen π_0, following residual RL (Silver et al., 2018; Ankile et al., 2025b); it is trained by behavior cloning and PPO on a chunk-level residual MDP, and a kNN trust gate limits it to the states it was trained on. The **calibrated novelty monitor** scores proprioception and image embeddings at 50 Hz; the score acts as an anomaly mask that switches the residual off, and, thresholded by split conformal prediction over consecutive control steps, it triggers escalation, which halts the skill with a bounded false-escalation rate per nominal call. The **typed agentic orchestrator** receives escalations in a state machine of LLM agents that plans, validates and risk-tags, requests human approval, executes and reviews protocol steps, and replans without repeating completed steps. We evaluate it on **AnomalyBio**, which we build on AutoBio. Figure 1 illustrates the pipeline. On *transfer*, the residual reaches 48.7% success against 40.7% for the published π_0 of Lan et al. (2026), an uncontrolled comparison across different checkpoints and episodes, and escalation detects 52% of monitor-reachable injected failures at 1.8% false escalations per call, although its protocol-level gain is within one standard error. Our contributions are:

- **Trust-gated chunk residual**: a chunk-level residual MDP for a frozen flow VLA, with a residual that sees the proposed chunk and a visual embedding and is restricted to its training support by a kNN trust gate (Secs. 3–4.4).
- **Calibrated novelty monitor**: one score, calibrated by split conformal prediction, that masks a learned residual at the control rate and escalates to a typed LLM replanner, with a bound on false escalations per nominal successful skill call (Sec. 4.3).
- **Typed agentic orchestrator**: a plan–validate–approve–execute–review state machine with deterministic routing and failure flags, and replanning that preserves completed protocol steps (Sec. 4.1).
- **AnomalyBio**: a benchmark built on the AutoBio simulator (Lan et al., 2026) for failure detection and recovery in laboratory protocols: multi-step protocols judged by simulator ground truth, a catalog of failures injected at chunk boundaries, and hazardous and compositionally infeasible goal probes, evaluated with single-component ablations, retry-only and oracle-reviewer baselines, and shadow-mode measurement of false escalations (Sec. 5, App. E).

2 Related Work

Correcting a frozen policy. Residual policy learning adds a learned term to a fixed controller (Silver et al., 2018; Johannink et al., 2019). ResiP (Ankile et al., 2025b) trains a per-step residual with proximal policy optimization (PPO) on a chunked diffusion policy, Policy Decorator (Yuan et al., 2025) bounds a SAC residual by $\alpha \tanh(\cdot)$, ResFiT (Ankile et al., 2025a) conditions an off-policy residual on the base action, and EXPO (Dong et al., 2026c;a) and Real-Time EXPO-FT (Dong et al., 2026b) learn edit policies, the latter editing actions within VLA chunks; DSRL (Wagenmaker et al., 2025) steers the base policy's latent noise. On VLAs, PLD (Xiao et al., 2026), HiL-ResRL (Liu et al., 2026a) and the policy-agnostic OmniTacTune (Yu et al., 2026) learn residual corrections, while VLA-Precision (Su et al., 2026) (LoRA), SimpleVLA-RL (Li et al., 2026), π_{RL} (Chen et al., 2025), VLA-RL (Lu et al., 2025) and RL4VLA (Liu et al., 2025a) fine-tune the whole model. The Residual Copilot (Sha et al., 2026) trains a PPO residual over teleoperation with a kNN model of the human pilot. These methods bound the *magnitude* of a correction; we also restrict *where* it acts and couple it to calibrated escalation. We do not claim residual RL on a frozen VLA as new.

Novelty-gated control and failure detection. Handing control away in novel states appears in shared autonomy (Reddy et al., 2018), gated imitation (Hoque et al., 2021; Menda et al., 2019), recovery policies (Thananjeyan et al., 2021) and safe fallbacks (Richter & Roy, 2017); deep kNN distance is a strong training-free out-of-distribution score (Sun et al., 2022). Sentinel (Agia et al., 2024), FAIL-Detect (Xu et al., 2025), SAFE (Gu et al., 2025) and FIPER (Römer et al., 2025) calibrate failure scores of generative and VLA policies by conformal prediction on successful rollouts,

as we do. Closest is Sinha et al. (2024): a fast embedding-based anomaly detector followed by LLM reasoning that selects a fallback. In our design the fast score also masks a learned controller at the control rate, and escalation enters a typed state machine that replans, re-tags risk and re-requests approval.

LLM planners, verifiers and laboratory automation. LLMs plan over skill libraries (Ahn et al., 2022; Huang et al., 2022; Liang et al., 2023) or delegate to symbolic planners (Liu et al., 2023a), but verify their own plans poorly (Kambhampati et al., 2024; Huang et al., 2024), so systems add VLM constraint monitors (Guo et al., 2024), failure reasoners (Liu et al., 2023b; Duan et al., 2025; Ye et al., 2025; Lin et al., 2026), plan filters (Obi et al., 2025; Ravichandran et al., 2026), conformal help-seeking (Ren et al., 2023), verifier agents over VLAs (Yang et al., 2025) or SOP-encoded roles (Hong et al., 2024). In the lab, Coscientist (Boiko et al., 2023) and ChemCrow (Bran et al., 2024) drive instruments through tools; ORGANA (Darvish et al., 2024) and BioMARS (Qiu et al., 2025), which has an inspector agent for anomaly detection, couple LLM agents with arms; BioProAgent (Liu et al., 2026b) anchors planning in a finite-state machine; BioProVLA-Agent (Du et al., 2026) combines a protocol agent, a VLM verifier and a VLA executor. None of these corrects the VLA inside a skill, and none uses a calibrated numerical monitor, with a bounded false-escalation rate, that also masks a learned residual. Lab simulators include AutoBio (Lan et al., 2026), LabUtopia (Li et al., 2025), Chemistry3D (Li et al., 2024) and Pipette (Liu et al., 2026c).

3 Problem Setting

Protocol level. A natural-language goal G must be achieved with a library $\mathcal{K}$ of typed skills exposed as tools with JSON argument schemas. A plan $P = (p_1, \dots, p_{|P|})$ is a list of *protocol steps*; each p_i carries a description, a skill name $\mathrm{sk} \in \mathcal{K}$, pre- and postconditions stated as checkable facts, an expected observation $\mathrm{obs}^\star$, and a risk tag $\mathrm{req} \in \{0, 1\}$ with a reason. The orchestrator state X (App. A) holds G, P, a step index i, a world state, an execution log, feedback, decisions and a coordinator budget counter. A deterministic router $R(X)$ chooses the next node, and LLM nodes write typed fields of X. Protocol success is judged by the simulator's task predicates, never by the system's own reviewer.

Execution level. A motion skill runs the frozen π_0 in an AutoBio scene with observation $o_t = (I_t, s_t, \ell)$ (images, proprioception $s_t \in \mathbb{R}^{d_s}$, instruction) and joint-space actions $a_t \in \mathbb{R}^{d_a}$, one *control step* per Δt=20 ms. At $t_j = jH$ the VLA returns a *chunk* $\mathbf{C}_j \in \mathbb{R}^{H \times d_a}$, H=50, executed open-loop for 1 s as in the AutoBio evaluator.

Chunk-level residual MDP. The residual makes one decision per chunk. At t_j it observes $x_j = [\tilde{s}_j, \tilde{\bar{c}}_j, \phi(I_{t_j})]$: normalized proprioception, the normalized mean row $\bar{c}_j = \frac{1}{H}\sum_h \mathbf{C}_{j,h}$ of the proposed chunk, and a frozen DINOv2 embedding (Oquab et al., 2024) of the chunk-start frame. It emits $u_j \in (-1, 1)^{d_a}$, which becomes a correction δ_j added to every row:

$$a_{t_j+h} \;=\; \mathbf{C}_{j,h} \;+\; m_{t_j+h}\, g_j\, \delta_j, \qquad h = 0, \dots, H-1, \tag{1}$$

where $g_j \in \{0\} \cup [g_{\min}, 1]$ is the trust gate (Eq. (3)) and $m_t \in \{0, 1\}$ the anomaly mask (Eq. (5)); $g_j = m_t = 1$ recovers ungated residual RL. We define the latent state $z_j = (x^{\mathrm{sim}}_{t_j}, \mathbf{C}_j, \bar{u}_{j-1}, t_j)$: simulator state, proposed chunk, previous *gated* normalized residual $\bar{u}_{j-1} = g_{j-1}u_{j-1}$, and elapsed time. π_0 is queried memorylessly (no image history, fresh flow noise per query, no recurrent state), so $\mathbf{C}_{j+1} \sim \pi_0(\cdot \mid o_{t_{j+1}})$ depends only on the next simulator state, and the process is Markov in z_j with a stationary kernel that executes Eq. (1) for up to H rows and then samples $\mathbf{C}_{j+1}$. The residual sees only x_j (a POMDP); the critic also receives $(\bar{u}_{j-1}, t_j/T)$. A 15 s *transfer* episode has at most 15 decisions.

4 Method

4.1 Orchestration: a typed state machine of LLM agents

The orchestrator is a LangGraph (LangChain, Inc., 2024) graph over one typed state X; nodes communicate only by overwriting typed fields, and every LLM output is a schema-typed structured

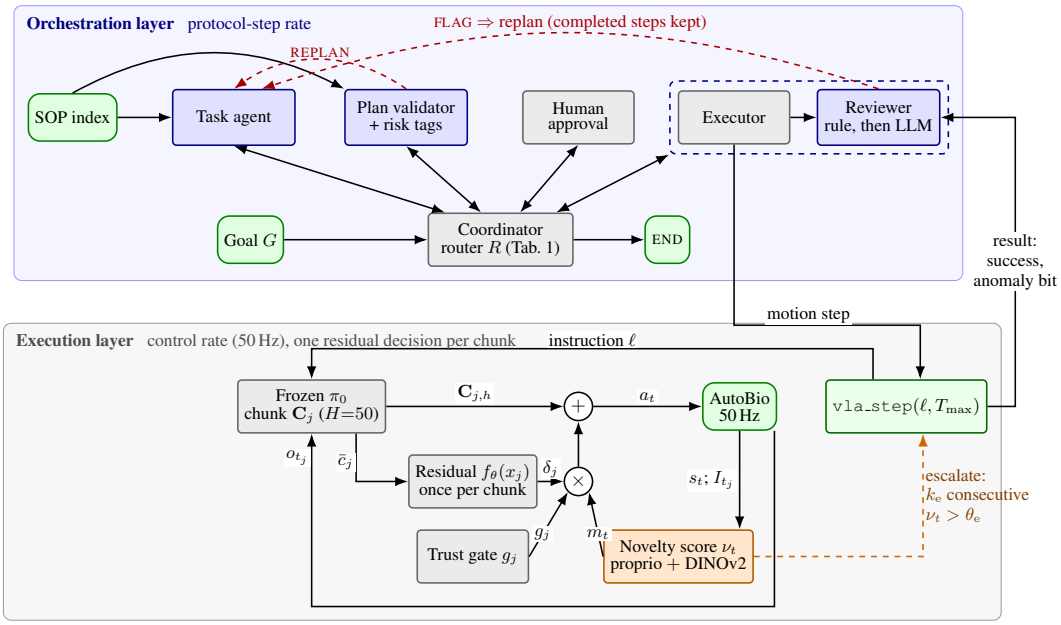

Figure 1: **ARGO-VLA.** Blue boxes are LLM calls, grey boxes deterministic code (the executor's argument parser is also an LLM call), green boxes inputs and outputs, orange the novelty score. *Top:* the SOP chunks go to the task agent and the validator; every node returns to the coordinator, which applies the router of Tab. 1; dashed red arrows are replanning loops that pass through the coordinator. *Bottom:* a motion step calls `vla_step`, which runs the frozen π_0 chunk by chunk. A residual computed once per chunk is scaled by the trust gate g_j and switched off at control steps where the anomaly mask $m_t{=}0$. If ν_t exceeds θ_e for k_e consecutive control steps, the skill halts and returns the anomaly bit, which forces a reviewer FLAG.

output. Tab. 1 gives the router and Alg. 2 (App. A) the node semantics. The **task agent** writes a plan of typed protocol steps whose skills must come from the tool list, grounded in the top-5 chunks of a standard-operating-procedure (SOP) index retrieved once per run. The **plan validator** is a single structured call that returns APPROVE/REPLAN/ABORT with feedback and per-step risk tags. Its prompt rules (App. B) tag hazardous reagents, a heater above 60 °C, vacuum, incompatible chemical mixes and irreversible disposal; a deterministic rule layer applies the same rules to typed tool arguments (e.g. `heat.temp_c` > 60), and a step is tagged if either layer tags it. Tagged steps need **human approval**; rejection aborts the run. The **executor** parses tool arguments from the step description with one structured LLM call and dispatches the tool over the Model Context Protocol (MCP) (Anthropic, 2024). The **reviewer** first applies a deterministic rule: if the tool result reports failure or carries the anomaly bit, the step is logged as failed and flagged without an LLM call. Otherwise an LLM compares the result and world state with $\mathrm{obs}^\star$ and the postconditions and returns PASS or FLAG. A flag reason becomes feedback, and the router sends the run to the task agent. *Replanning preserves completed steps*: the task agent receives the ids of steps already reviewed as PASS and a summary of the full execution log and plans only the remaining work, which is appended to the completed steps. A deterministic check rejects a plan that re-emits a completed irreversible step, and all human approvals are cleared, so every tagged step of the new plan is approved again. A coordinator budget $B{=}80$ visits, which also bounds replanning loops, ends a run with `abort`. Each planning round costs two LLM calls and each executed protocol step at most two.

The `vla_step` skill. A motion step invokes $\texttt{vla_step}(\ell, T_{\max})$ with a natural-language instruction and a time cap. Alg. 1 specifies it: it runs the gated residual stack from the *current* simulator state until the success predicate holds, $T_{\max}$ (capped by the AutoBio time limit) elapses, or the monitor escalates. It returns $\{\text{success}, \text{anomaly}, \text{trace summary}, \text{predicates}\}$. The scene is never reset after an escalation, so a replanned protocol resumes from the post-failure state.

Table 1: Router $R(X)$, evaluated top to bottom (left block, then right); first match wins.

#	condition on X	next node	#	condition on X	next node
0	budget $> B$	abort, END	5	decision = ABORT†	END
1	status $\in$ {success, abort}	END	6	decision = REPLAN*	task agent
2	operator interrupt	validator	7	$i \geq \lvert P \rvert$	END
3	P=NONE $\vee$ feedback	task agent	8	$\mathrm{req}(p_i) \wedge p_i$ unapproved	approval
4	no validator decision	validator	9	otherwise	exec.+review

* Subsumed by rule 3: the validator node always sets feedback on REPLAN. † Subsumed by rule 1: ABORT sets status = abort.

4.2 GATED RESIDUAL EXECUTION

Bounded residual in normalized units. With $\tilde{s}$ the state standardized by dataset mean and standard deviation, and $w \in \mathbb{R}^{d_a}_{>0}$ the per-dimension range (1st to 99th percentile) of the training actions, the residual is

$$\delta_j = \alpha\, w \odot u_j, \qquad u_j = \tanh f_\theta(x_j), \qquad |\delta_{j,d}| < \alpha\, w_d, \tag{2}$$

where f_θ is an MLP with 48 ReLU units and a zero-initialized last layer, so an untrained head reproduces the VLA exactly (Silver et al., 2018). We use α=0.08, i.e. 8% of each dimension's range; ResiP scales a $[-1, 1]$-normalized residual by 0.1, i.e. 5% of the range (Ankile et al., 2025b). Normalization matters because AutoBio mixes radians with a $[0, 255]$ gripper command.

Trust gate. Let $\mathcal{D}$ be the proprioceptive states on which the residual is trained: chunk-start states of the log-only π_0 rollouts (Alg. 3, line 1) plus expert states; $\mathcal{D}$ is frozen before BC and PPO. With $\bar{d}_K(s)$ the mean Euclidean distance from s to its K=10 nearest neighbors in $\mathcal{D}$ (excluding s itself when $s \in \mathcal{D}$),

$$g_j = \begin{cases} e^{-\bar{d}_K(s_j)/\beta} & \text{if } e^{-\bar{d}_K(s_j)/\beta} \geq g_{\min}, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

with β=0.12 and $g_{\min}$=0.15. For 7-D single-arm states (six joint angles and the gripper joint, all in radians) the trust gate turns off at $\bar{d}_K = \beta \ln(1/g_{\min}) \approx 0.23$ rad; β is set per task for 14-D bimanual states. The buffer is built from the residual's training states, not from the demonstrations alone, because the trust gate certifies the residual's own support: a correction learned from data is trusted only near that data, as in novelty-based handoff (Hoque et al., 2021; Sun et al., 2022).

The trust gate switches off where the VLA may fail. Off-support, the residual's output is an extrapolation, so the trust gate bounds the residual to its support; states outside it are the job of the anomaly channel, which masks any remaining correction and escalates persistent novelty. An ablation retrains the residual without the trust gate.

4.3 NOVELTY SCORE, CONFORMAL CALIBRATION AND ESCALATION

Phase-conditioned novelty score. At control step t inside chunk j,

$$\nu_t \;=\; \frac{w_p}{\kappa_p} \left\| (s_t - \mu_j) \oslash \sigma_j \right\|_2 \;+\; \frac{w_v}{\kappa_v}\, \bar{d}^{\mathcal{M}}_{K_v}(e_j), \tag{4}$$

where (μ_j, σ_j) are the mean and standard deviation of proprioception over the control steps of chunk index j across a statistics set $\mathcal{R}_{\mathrm{stat}}$ of nominal successful π_0-only rollouts, accumulated with Welford's single-pass update (Welford, 1962), with a variance floor $\sigma_j^2 \geq 10^{-4}$ and the last index reused beyond the longest rollout. Conditioning on the phase j makes the score measure deviation from nominal behavior at the current point of the skill rather than distance from the task's average pose. The visual term is the mean distance from the ℓ_2-normalized DINOv2 embedding e_j of the chunk-start frame to its K_v=5 nearest neighbors in a bank $\mathcal{M}$ of frames from $\mathcal{R}_{\mathrm{stat}}$; it is constant within a chunk. The constants κ_p, κ_v are the medians of each term on $\mathcal{R}_{\mathrm{stat}}$ (the visual term leave-one-out, excluding a frame's own entry in $\mathcal{M}$), so $(w_p, w_v) = (0.7, 0.3)$ are relative weights. All monitor statistics are *frozen* at deployment; there is no online update.

Algorithm 1 Anomaly-masked residual chunk execution (`vla_step`)

Input: ℓ, $T_{\max}$, frozen π_0, f_θ, buffer $\mathcal{D}$, frozen monitor $(\mu_j, \sigma_j, \mathcal{M}, \kappa_p, \kappa_v)$, $\theta_{\mathrm{m}}, \theta_{\mathrm{e}}, k_{\mathrm{e}}$, flag SHADOW

$c \leftarrow 0$; $j \leftarrow 0$; esc $\leftarrow$ FALSE ▷ t, j count from the call's first control step
while not SUCCESS(x^{sim}) **and** $jH\Delta t < T_{\max}$ **do** ▷ checked between chunks, as in AutoBio
  $o \leftarrow$ OBSERVE(); $\mathbf{C}_j \sim \pi_0(\cdot \mid o, \ell)$; $e_j \leftarrow \phi(o.I)$ ▷ e_j shared by residual and monitor
  $g_j \leftarrow$ Eq. (3); $\delta_j \leftarrow$ Eq. (2) with $x_j = [\tilde{s}_j, \tilde{\mathbf{c}}_j, e_j]$ ▷ training: $u_j = \tanh y_j$, $y_j \sim \mathcal{N}(f_\theta, \varsigma^2)$
  for $h = 0, \ldots, H-1$ **do**
    $t \leftarrow jH + h$; read s_t; $\nu_t \leftarrow$ Eq. (4); $m_t \leftarrow \mathbf{1}[\nu_t \le \theta_{\mathrm{m}}]$; apply $a_t = \mathbf{C}_{j,h} + m_t g_j \delta_j$ for one control step
    $c \leftarrow c + 1$ **if** $\nu_t > \theta_{\mathrm{e}}$ **else** 0
    if $c \ge k_{\mathrm{e}}$ **and not** esc **then**
      esc $\leftarrow$ TRUE; log escalation at t
      if not SHADOW **then**
        return {FALSE, anomaly=TRUE, trace, predicates}
      end if
    end if
  end for
  $j \leftarrow j + 1$
end while
return {SUCCESS(x^{sim}), anomaly=FALSE, trace (incl. shadow esc), predicates}

Split conformal calibration. For scores $S_1, \ldots, S_n$ and miscoverage ϵ, let $\hat{\theta}(\epsilon) = S_{(\lceil (n+1)(1-\epsilon) \rceil)}$, the $\lceil (n+1)(1-\epsilon) \rceil$-th smallest score, with $\hat{\theta} = +\infty$ if $n < 1/\epsilon - 1$ (Angelopoulos & Bates, 2021). If a new score is exchangeable with the n calibration scores, then marginally over calibration and test draws $\Pr[S_{\mathrm{test}} > \hat{\theta}(\epsilon)] \le \epsilon$. The mask threshold targets a per-control-step mask rate on nominal data:

$$m_t = \mathbf{1}[\nu_t \le \theta_{\mathrm{m}}], \qquad \theta_{\mathrm{m}} = \hat{\theta}(\epsilon_{\mathrm{m}}) \text{ on pooled control-step scores}, \tag{5}$$

computed on a split $\mathcal{R}_{\mathrm{mask}}$ of nominal successful π_0-only rollouts, disjoint from $\mathcal{R}_{\mathrm{stat}}$. Control steps within an episode are dependent, and $\mathcal{R}_{\mathrm{mask}}$ precedes the residual (Alg. 3), so θ_{m} carries no formal guarantee; the realized mask rate is that of the deployed pipeline on $\mathcal{R}_{\mathrm{cal}}$. The *escalation* threshold carries the guarantee. We escalate when $\nu_t > \theta_{\mathrm{e}}$ for k_{e} consecutive control steps. Here an *episode* is one `vla_step` call, and t and j count from its first control step; escalation happens in a call if and only if

$$E = \max_{t_0 + k_{\mathrm{e}} - 1 \le t \le t_{\mathrm{end}}} \min_{t - k_{\mathrm{e}} < t' \le t} \nu_{t'} \; > \; \theta_{\mathrm{e}}, \qquad \theta_{\mathrm{e}} = \hat{\theta}(\epsilon_{\mathrm{e}}) \text{ on } \{E_i\}_{i=1}^{n_{\mathrm{cal}}}, \tag{6}$$

where t_0 and t_{end} are the first and last control steps of the call and the E_i come from a calibration set $\mathcal{R}_{\mathrm{cal}}$ of n_{cal} nominal successful episodes of the *full deployed pipeline* (mask on, escalation in shadow mode), disjoint from $\mathcal{R}_{\mathrm{stat}}$. The guarantee is therefore marginal: for a new nominal call exchangeable with $\mathcal{R}_{\mathrm{cal}}$, i.e. a nominal call that would succeed, $\Pr[\text{false escalation}] \le \epsilon_{\mathrm{e}}$ per `vla_step` call. It says nothing about calls that fail without injected faults, and does not cover a call resumed from a post-failure state after a replan, which is not exchangeable with $\mathcal{R}_{\mathrm{cal}}$. If all q `vla_step` calls actually made in a protocol (including retries) are nominal and would succeed, the protocol's false-escalation probability is at most $q\,\epsilon_{\mathrm{e}}$ by the union bound; on escalation the skill halts mid-chunk and the reviewer rule forces a replan (Sec. 4.1).

4.4 TRAINING THE RESIDUAL

Behavior cloning (BC) on chunk-mean deltas. Because δ_j is a constant offset over the chunk, the target is the chunk mean of the expert-minus-VLA difference. Reference actions come from *nearest-state relabeling*: for each logged chunk-start state, we take the nearest state (in proprioception plus object-pose space) among scripted-expert rollouts on the same scene seed, and the expert's next H actions from that state (AutoBio's experts cannot be re-planned from an arbitrary mid-episode state). With $\tilde{\delta}_j^\star = \mathrm{clip}\big(\frac{1}{H}\sum_h (a_{j,h}^{\mathrm{ref}} - \mathbf{C}_{j,h}) \oslash (\alpha w),\, -1 + \epsilon_0,\, 1 - \epsilon_0\big)$ and $\epsilon_0 = 10^{-3}$, we minimize

$$\mathcal{L}_{\mathrm{BC}}(\theta) = \mathbb{E}_{j:\, g_j > 0} \big\| \tanh f_\theta(x_j) - \tilde{\delta}_j^\star \big\|_2^2. \tag{7}$$

The target is not divided by g_j, so near the support edge the applied correction $g_j \delta_j$ is deliberately shrunk.

Table 2: Success rate (%) on AutoBio. *Published* columns are measured values copied from Tab. 4 of Lan et al. (2026) (RDT: Liu et al., 2025b; $\pi_{0.5}$: Black et al., 2025a), 100 demonstrations, mean $\pm$ SEM over 3 fine-tuning seeds; they are not re-run. The *VLA* column repeats the published π_0 values (same fine-tuning recipe and evaluator). The +R rungs start from one π_0 checkpoint per task fine-tuned with that recipe; success over 3 residual seeds $\times$ 100 episodes, $\pm$ binomial standard error of the pooled rate (n=300). All +R rungs use the trust gate (T); +R(RL) is evaluated with the anomaly mask off, +R(RL)+M with it on (the training condition); $-$T (retr.) is a residual retrained and evaluated without the trust gate. Thermal mixer (progress-scored) excluded.

	Published			π_0 base (VLA: published)				
Task (level)	π_0	RDT	$\pi_{0.5}$	VLA	+R(BC)	+R(RL)	+R(RL)+M	$-$T (retr.)
Close thermal-cycler lid (E)	99.7±0.3	100.0±0.0	99.4±0.3	99.7±0.3	99.3±0.5	99.7±0.3	99.7±0.3	99.7±0.3
Open thermal-cycler lid (E)	96.0±2.1	99.0±0.6	95.6±1.4	96.0±2.1	95.0±1.3	96.0±1.1	96.0±1.1	95.3±1.2
Pick up centrifuge tube (E)	53.7±5.9	57.7±1.2	95.9±0.9	53.7±5.9	53.0±2.9	59.7±2.8	60.0±2.8	58.7±2.8
Unscrew tube cap (M)	21.3±1.5	2.7±1.2	11.7±1.1	21.3±1.5	20.7±2.3	23.0±2.4	23.3±2.4	22.7±2.4
Aspirate with pipette (M)	42.7±1.8	0.3±0.3	18.3±1.0	42.7±1.8	42.0±2.8	46.7±2.9	47.0±2.9	45.7±2.9
Transfer centrifuge tube (M)	40.7±5.4	2.0±1.2	37.3±11.3	40.7±5.4	41.7±2.8	48.7±2.9	49.0±2.9	47.7±2.9
Screw on tube cap (H)	2.0±0.6	8.3±4.4	1.7±0.2	2.0±0.6	1.7±0.7	2.7±0.9	2.7±0.9	2.7±0.9
Load centrifuge rotor (H)	14.7±1.3	1.0±1.0	16.0±0.3	14.7±1.3	14.7±2.0	18.7±2.3	19.0±2.3	17.7±2.2

PPO on the chunk-level MDP. The policy is Gaussian before squashing, $y_j \sim \mathcal{N}(f_\theta(x_j), \operatorname{diag}\varsigma^2)$ with $u_j = \tanh y_j$, the VLA frozen and both the trust gate and the anomaly mask on (escalation off). Using the success, termination, magnitude and smoothness terms of Sha et al. (2026) on the *gated* residual $\bar{u}_j = g_j u_j$ (the mask is left out, so the reward does not depend on the monitor),

$$\begin{aligned} r_j = R_s \mathbf{1}[\text{first success at end of chunk } j] - R_f \mathbf{1}[\text{timeout} \vee \text{unhealthy}] \\ - \lambda_{\text{mag}} \|\bar{u}_j\|_2 - \lambda_{\text{jerk}} \|\bar{u}_j - \bar{u}_{j-1}\|_2, \end{aligned} \tag{8}$$

so a chunk with g_j=0 has a reward independent of u_j, and we drop it from the policy term (it still provides value targets). *Unhealthy* is the environment's invalid-state termination. With termination indicators d_j and generalized advantage estimation (GAE) (Schulman et al., 2016) (segments truncated before termination are bootstrapped from V_ψ), $\hat{A}_j = \sum_{l \geq 0} (\gamma\lambda)^l \prod_{i<l} (1 - d_{j+i}) \left(r_{j+l} + \gamma(1 - d_{j+l}) V_\psi(\omega_{j+l+1}) - V_\psi(\omega_{j+l}) \right)$ with critic input $\omega_j = (x_j, \bar{u}_{j-1}, t_j/T)$, we maximize

$$\mathcal{L}^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}_j \Big[\min \left(\varrho_j \hat{A}_j,\ \text{clip}(\varrho_j, 1 - c, 1 + c) \hat{A}_j \right) \Big], \qquad \varrho_j = \frac{\pi_\theta(y_j \mid x_j)}{\pi_{\theta_{\text{old}}}(y_j \mid x_j)}, \tag{9}$$

the clipped PPO objective (Schulman et al., 2017); the ratio over y_j equals the ratio over u_j because the $\tanh$ Jacobian does not depend on θ and cancels in the ratio. With at most 15–45 decisions per episode, γ=0.995 per chunk keeps the return close to undiscounted ($0.995^{45} \approx 0.80$ on the longest task), so the terminal success reward dominates. Alg. 3 (App. C) orders the stages: log-only rollouts, monitor statistics and mask threshold, BC, PPO with the mask on, and finally the escalation threshold on the final pipeline.

5 Experiments

Benchmark and seeds. We use AutoBio (Lan et al., 2026) (MuJoCo, 50 Hz control, 100 scripted demonstrations per task; Fig. 2). For each task we fine-tune one π_0 checkpoint with the AutoBio recipe in OpenPI (the π_0 checkpoint `pi0_base`, not $\pi_{0.5}$; 30k steps, batch 32, 100 demonstrations). Each residual rung is trained with 3 seeds and evaluated on 100 episodes per seed with the same episode seeds for all methods; we report the pooled success rate over the 300 episodes $\pm$ its binomial standard error $\sqrt{p(1-p)/300}$, which captures episode sampling but not seed-to-seed variance. The VLA baseline is AutoBio's published π_0 result for the same recipe and evaluator, not a re-run; its SEM is over three π_0 fine-tunes. The target tasks are the eight success-scored tasks of Tab. 2 (App. D), with *transfer centrifuge tube* (UR5e, 7-D) as the primary task. We chose π_0 as the frozen base from five candidate base policies because it has the lowest first-action error, the substrate that a residual correcting small systematic errors needs (App. D, Tab. 7).

Table 3: Protocol level on AnomalyBio under injected failures. Injection rate ρ=0.2, 150 runs per row (binomial SE $\approx$3–4 points for success). Success, recovery and detection in %, judged by simulator ground truth; latency is the median in seconds; false escalation is the % of nominal successful `vla_step` calls that escalate in shadow mode. Rows (i)–(v) and (iv+v) ablate the full system. * Assumes θ_e recalibrated per variant.

	Succ. ↑	Recov. ↑	Detect. ↑	Latency (s) ↓	False esc. ↓	Approvals/run
Open-loop	16.7	—	—	—	—	1.15
Retry-only	28.0	12.4	—	—	—	1.15
Agentic	45.3	37.8	—	—	—	1.50
Agentic+E (full)	48.0	40.5	52	0.5	1.8	1.55
Oracle reviewer	49.3	42.4	52	0.5	1.8	1.50
(i) no validator critique	47.3	39.7	52	0.5	1.8	1.30
(ii) deterministic reviewer	48.0	40.1	52	0.5	1.8	1.50
(iii) mask on, no escalation	45.3	38.1	—	—	—	1.50
(iv) no trust gate (retr.)	47.3	39.2	52	0.5	1.9*	1.55
(v) no anomaly mask	47.3	39.9	53	0.5	1.8	1.55
(iv+v) neither	46.0	38.3	53	0.5	2.0*	1.55

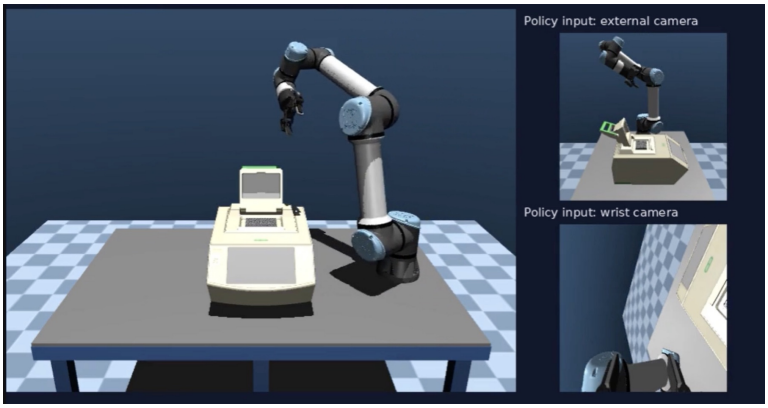

Figure 2: An AutoBio scene used in AnomalyBio: a UR5e at the thermal cycler (MuJoCo). *Left:* scene view. *Right:* the two camera images the frozen π_0 receives, external (top) and wrist (bottom); the novelty score's visual term embeds the same images.

Execution ladder and ablations. We compare **+R(BC)**, **+R(RL)** and **+R(RL)+M** on matched episode seeds, against the published frozen-π_0 result (**VLA**). Every +R rung keeps the trust gate it was trained with; +R(RL) is evaluated with the anomaly mask off, +R(RL)+M with it on (the PPO condition). Removing the trust gate means *retraining* without it (**−T (retr.)**), so no rung sees a gate setting it was not trained with. Fig. 3 (App. F) plots the ladder per task.

Protocol-level evaluation on AnomalyBio. AnomalyBio is our protocol-level benchmark on top of AutoBio. Its protocols chain AutoBio skills and non-motion tools (heat, mix, pipette) with goals written by us; skills sharing a scene run in one continuous episode, and success is judged only by simulator ground truth. We compare **open-loop** (one plan, no review), **retry-only** (a failed motion step is re-run up to three times, no LLM replanning), **agentic** (full state machine, monitor entirely off, residual +R(RL)) and **agentic+E** (full system: +R(RL)+M with escalation), plus an **oracle reviewer** that replaces the whole reviewer (rule and LLM) of agentic+E with the simulator's task predicates. Open-loop and retry-only use the agentic execution stack (+R(RL), monitor off) and the deterministic risk-tag rules without the validator's LLM call; retry-only detects a failure from the tool result. Each ablation of the full system changes one component, except (iv+v): (i) validator critique off, deterministic hazard tagging on; (ii) LLM reviewer replaced by a deterministic outcome check; (iii) monitor and mask on, escalation off (nothing halts or flags), which isolates escalation from masking; (iv) trust gate off, residual retrained without it; (v) anomaly mask off; and (iv+v) both off, since each mechanism may cover for the other. A scripted operator approves a tagged step

if and only if its hazard label is false. The injection rate ρ is the probability that a motion step receives one failure, drawn uniformly from F1–F5 and applied at a uniformly chosen chunk boundary, independently per step. Injected failures are observable at chunk boundaries: a tube displaced by 2–5 cm, a dropped tube, a tool error returned over MCP, wrong labware placed, and camera occlusion lasting at least two chunks. Goal probes test the validator: hazardous goals, and goals feasible skill by skill but infeasible in composition, which the deterministic checks cannot reject (App. E). **Metrics:** protocol success; recovery (injected failures after which the protocol still succeeds without human help); detection rate and latency (injection to escalation) over *monitor-reachable* injections (F1, F2 and F5, which change proprioception or images; F3 and F4 are handled by the reviewer and validator); false-escalation rate in *shadow mode* (escalations logged, not executed) on separate non-injected runs, per `vla_step` call; and approvals per run.

Hypotheses. **H1**: +R(RL) beats the VLA on tolerance-limited tasks. **H2**: retrained without the trust gate, large α hurts. **H3**: agentic+E recovers more protocols than agentic and retry-only at false escalation $\leq \epsilon_{\mathrm{e}}$ per call. **H4**: removing critique increases failures on compositionally infeasible goals.

Results. Tab. 2 gives the execution ladder. +R(BC) stays within one point of the published π_0 (VLA), since the VLA's first-action error is far below the residual bound and BC targets are therefore near zero. Compared with AutoBio's published π_0, which uses different checkpoints and episodes and is therefore not a controlled baseline, +R(RL) is 8.0 points higher on transfer, 6.0 on pick-up, 4.0 on aspirate and 4.0 on load rotor, at most 1.7 higher on the cap tasks and equal on the lids; the anomaly mask (at most +0.3) and retraining without the trust gate (at most -1.0) stay within one standard error. At the protocol level (Tab. 3; Fig. 4 in App. F) the success is 16.7% open-loop, 28.0% retry-only, 45.3% agentic, 48.0% agentic+E and 49.3% with an oracle reviewer. Escalation detects 52% of monitor-reachable injections with a median latency of 0.5 s at 1.8% false escalations per call, within ϵ_{e}=2%. The differences between agentic and agentic+E, and between each ablation and the full system, are below the 3–4-point standard error of each rate at 150 runs ($\approx$6 points for a difference), so we report them as no detectable effect; validator critique matters mainly on the G2 probes (55% rejected by the full system vs. 5% by ablation (i)). In sum, H4 holds and H1 is not tested in a controlled comparison; H3 holds against retry-only but not detectably against agentic; H2 is tested only at α=0.08, where retraining without the trust gate costs at most 1.0 point (Tab. 2).

6 Limitations

All evaluation is in simulation; liquids are quasi-static and cross-scene protocols carry state symbolically. The reviewer's LLM path reads privileged simulator predicates, which a VLM verifier would replace on hardware. One residual decision per 1 s chunk limits reactivity, and the residual cannot fix language-grounding errors. The mask threshold has no formal guarantee; the escalation bound is marginal over nominal successful calls and says nothing about missed detections. Risk tags can miss hazards the rules do not name, so human approval is a safeguard, not a certificate.

7 Conclusion

In ARGO-VLA one conformally calibrated novelty score masks a trust-gated residual on a frozen π_0 and escalates persistent anomalies, with a bounded false-escalation rate per call, to a typed LLM state machine that replans without repeating completed steps. On tolerance-limited tasks it scores up to 8 points above AutoBio's published π_0, an uncontrolled comparison; the protocol-level gain from escalation (45.3% to 48.0%) is not yet detectable.

GENERATIVE AI USAGE DISCLOSURE

During the preparation of this manuscript, the authors used generative AI tools to assist with drafting, restructuring, and improving the clarity of portions of the text, for implementing methods (coding assistance), for checking the consistency of equations, algorithms and reported numbers, and for plotting figures from the reported tables. All AI-assisted material was reviewed and verified by the authors before inclusion. Generative AI tools were not used to design the core methodology or to generate or alter experimental results. The authors take full responsibility for the final content, methodology, data, results, and conclusions presented in this manuscript.

REPRODUCIBILITY STATEMENT

Sec. 4 specifies every component with equations and algorithms. App. A gives the state schema, routing and budget semantics, App. B the system prompts, App. C the hyperparameters and notation, App. D the task definitions, and App. E the AnomalyBio failure-injection catalog. Experiments use the public AutoBio simulator and OpenPI code. We will release the orchestration package, the residual and gating stack, the monitor, the evaluation harness with fixed episode seeds, the AnomalyBio protocol suite and trained residual weights upon publication. Compute (one host, two NVIDIA RTX A5000 GPUs) is detailed in App. C and App. D.

ETHICS STATEMENT

Laboratory automation can involve hazardous materials. The design requires human approval for steps tagged as hazardous and escalates anomalies rather than acting through them. Risk tagging can be incomplete, so deployment on real equipment requires institutional safety review, physical interlocks and SOPs that encode local hazards. All experiments are in simulation; no human subjects or personal data are involved. Automating experimental biology is dual-use; we release no protocols for hazardous agents.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Zi-ang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking Failure Modes of Generative Policies: Runtime Monitoring of Consistency and Progress. In *Conference on Robot Learning (CoRL)*, 2024. arXiv:2410.04640.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Conference on Robot Learning (CoRL)*, 2022. arXiv:2204.01691.

Anastasios N. Angelopoulos and Stephen Bates. A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. arXiv preprint arXiv:2107.07511, 2021.

Lars Ankile, Zhenyu Jiang, Rocky Duan, Guanya Shi, Pieter Abbeel, and Anusha Nagabandi. Residual Off-Policy RL for Finetuning Behavior Cloning Policies. arXiv preprint arXiv:2509.19301, 2025a.

Lars Ankile, Anthony Simeonov, Idan Shenfeld, Marcel Torne, and Pulkit Agrawal. From Imitation to Refinement – Residual RL for Precise Assembly. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025b. arXiv:2407.16677.

Anthropic. Model Context Protocol Specification (rev. 2024-11-05). `https://modelcontextprotocol.io/specification/2024-11-05`, 2024.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. In *Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Robotics: Science and Systems (RSS)*, 2025b. arXiv:2410.24164.

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023. DOI 10.1038/s41586-023-06792-0.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024. arXiv:2304.05376.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online RL Fine-tuning for Flow-based Vision-Language-Action Models. arXiv preprint arXiv:2510.25889, 2025.

Kourosh Darvish, Marta Skreta, Yuchi Zhao, Naruki Yoshikawa, Sagnik Som, Miroslav Bogdanovic, Yang Cao, Han Hao, Haoping Xu, Alán Aspuru-Guzik, Animesh Garg, and Florian Shkurti. ORGANA: A Robotic Assistant for Automated Chemistry Experimentation and Characterization. arXiv preprint arXiv:2401.06949, 2024.

Perry Dong, Kuo-Han Hung, Tian Gao, Dorsa Sadigh, and Chelsea Finn. EXPO-FT: Sample-Efficient Reinforcement Learning Finetuning for Vision-Language-Action Models. arXiv preprint arXiv:2605.25477, 2026a.

Perry Dong, Kuo-Han Hung, Dorsa Sadigh, and Chelsea Finn. Reinforcement Learning for Real-Time Vision-Language-Action Policies. arXiv preprint arXiv:2609.18207, 2026b.

Perry Dong, Qiyang Li, Dorsa Sadigh, and Chelsea Finn. EXPO: Stable Reinforcement Learning with Expressive Policies. In *International Conference on Learning Representations (ICLR)*, 2026c.

Zhaohui Du, Zhe Wang, Dongzhan Zhou, Minting Pan, Hongmei Fei, Xiwen Cao, Ting Xiao, Qi Wang, Huanbo Jin, Jiaming Gu, Quan Lu, and Zhe Liu. BioProVLA-Agent: An Affordable, Protocol-Driven, Vision-Enhanced VLA-Enabled Embodied Multi-Agent System with Closed-Loop-Capable Reasoning for Biological Laboratory Manipulation. arXiv preprint arXiv:2605.07306, 2026.

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 (ICLR)*, 2025.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. SAFE: Multitask Failure Detection for Vision-Language-Action Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2506.09937.

Yanjiang Guo, Yen-Jen Wang, Lihan Zha, and Jianyu Chen. DoReMi: Grounding Language Model by Detecting and Recovering from Plan-Execution Misalignment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024. arXiv:2307.00329.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *International Conference on Learning Representations (ICLR)*, 2024.

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 *Conference on Robot Learning (CoRL)*, 2021. arXiv:2109.08273.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.01798.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner Monologue: Embodied Reasoning through Planning with Language Models. In *Conference on Robot Learning (CoRL)*, 2022. arXiv:2207.05608.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual Reinforcement Learning for Robot Control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. arXiv:1812.03201.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. LLMs Can't Plan, But Can Help Planning in LLM-Modulo Frameworks. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2402.01817.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Conference on Robot Learning (CoRL)*, 2024. arXiv:2406.09246.

Zhiqian Lan, Yuxuan Jiang, Ruiqi Wang, Xuanbing Xie, Rongkui Zhang, Yicheng Zhu, Peihang Li, Tianshuo Yang, Tianxing Chen, Haoyu Gao, Xiaokang Yang, Xuelong Li, Hongyuan Zhang, Yao Mu, and Ping Luo. AutoBio: A Simulation and Benchmark for Robotic Automation in Digital Biology Laboratory. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2505.14030.

LangChain, Inc. LangGraph. `https://github.com/langchain-ai/langgraph`, 2024. Software; version pinned in the released code, accessed September 2026.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA Training via Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.09674.

Rui Li, Zixuan Hu, Wenxi Qu, Jinouwen Zhang, Zhenfei Yin, Sha Zhang, Xuantuo Huang, Hanqing Wang, Tai Wang, Jiangmiao Pang, Wanli Ouyang, Lei Bai, Wangmeng Zuo, Ling-Yu Duan, Dongzhan Zhou, and Shixiang Tang. LabUtopia: High-Fidelity Simulation and Hierarchical Benchmark for Scientific Embodied Agents. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2025. arXiv:2505.22634.

Shoujie Li, Yan Huang, Changqing Guo, Tong Wu, Jiawei Zhang, Linrui Zhang, and Wenbo Ding. Chemistry3D: Robotic Interaction Benchmark for Chemistry Experiments. arXiv preprint arXiv:2406.08160, 2024.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023. arXiv:2209.07753.

Zijun Lin, Jiafei Duan, Haoquan Fang, Dieter Fox, Ranjay Krishna, Cheston Tan, and Bihan Wen. FailSafe: Reasoning and Recovery from Failures in Vision-Language-Action Models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2026. arXiv:2510.01642.

Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency. arXiv preprint arXiv:2304.11477, 2023a.

Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What Can RL Bring to VLA Generalization? An Empirical Study. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a. arXiv:2505.19789.

Jingyi Liu, Zhaohong Mai, ShunSen He, Hang Ren, Chao Wang, Shunbo Zhou, XiaoDong Wu, and Heng Zhang. HiL-ResRL: A Model-Agnostic Finetuning Adapter via Human-in-the-loop Residual Reinforcement Learning. arXiv preprint arXiv:2606.22860, 2026a.

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 (ICLR)*, 2025b. arXiv:2410.07864.

Yuyang Liu, Jingya Wang, Liuzhenghao Lv, and Yonghong Tian. BioProAgent: Neuro-Symbolic Grounding for Constrained Scientific Planning. arXiv preprint arXiv:2603.00876, 2026b.

Zeyi Liu, Arpit Bahety, and Shuran Song. REFLECT: Summarizing Robot Experiences for Failure Explanation and Correction. In *Conference on Robot Learning (CoRL)*, 2023b. arXiv:2306.15724.

Zhe Liu, Huanbo Jin, Zhaohui Du, Zhe Wang, Dongzhan Zhou, Minting Pan, He Xu, Peijia Li, Jiaming Gu, Quan Lu, Qi Wang, Bin Ji, and Ting Xiao. Pipette: An Embodied Simulation Platform, Benchmark, and Data-Efficient Augmentation Framework for Wet-Lab Robotics. arXiv preprint arXiv:2606.12936, 2026c.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. VLA-RL: Towards Masterful and General Robotic Manipulation with Scalable Reinforcement Learning. arXiv preprint arXiv:2505.18719, 2025.

Kunal Menda, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. EnsembleDAgger: A Bayesian Approach to Safe Imitation Learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019. arXiv:1807.08364.

NVIDIA. Cosmos 3: Omnimodal World Models for Physical AI. Technical report, NVIDIA Research, June 2026. URL `https://research.nvidia.com/labs/cosmos-lab/cosmos3/technical-report.pdf`. arXiv:2606.02800. Cosmos 3 Edge (4B) model card: `https://huggingface.co/nvidia/Cosmos3-Edge`.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. arXiv preprint arXiv:2503.14734, 2025.

Ike Obi, Vishnunandan L. N. Venkatesh, Weizheng Wang, Ruiqi Wang, Dayoon Suh, Temitope I. Amosa, Wonse Jo, and Byung-Cheol Min. SafePlan: Leveraging Formal Logic and Chain-of-Thought Reasoning for Enhanced Safety in LLM-based Robotic Task Planning. arXiv preprint arXiv:2503.06892, 2025.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research (TMLR)*, 2024. arXiv:2304.07193.

E. S. Page. Continuous Inspection Schemes. *Biometrika*, 41(1/2):100–115, 1954.

Yibo Qiu, Zan Huang, Zhiyu Wang, Handi Liu, Yiling Qiao, Yifeng Hu, Shu'ang Sun, Hangke Peng, Ronald X. Xu, and Mingzhai Sun. BioMARS: A Multi-Agent Robotic System for Autonomous Biological Experiments. arXiv preprint arXiv:2507.01485, 2025.

Qwen Team. Qwen3 Technical Report. arXiv preprint arXiv:2505.09388, 2025.

Zachary Ravichandran, Alexander Robey, Vijay Kumar, George J. Pappas, and Hamed Hassani. Safety Guardrails for LLM-Enabled Robots. *IEEE Robotics and Automation Letters*, 11:4649–4656, 2026. arXiv:2503.07885.

Siddharth Reddy, Anca D. Dragan, and Sergey Levine. Shared Autonomy via Deep Reinforcement Learning. In *Robotics: Science and Systems (RSS)*, 2018. arXiv:1802.01744.

Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots That Ask For Help: Uncertainty Alignment for Large Language Model Planners. In *Conference on Robot Learning (CoRL)*, 2023. arXiv:2307.01928.

Charles Richter and Nicholas Roy. Safe Visual Navigation via Deep Learning and Novelty Detection. In *Robotics: Science and Systems (RSS)*, 2017.

Ralf Römer, Adrian Kobras, Luca Worbis, and Angela P. Schoellig. Failure Prediction at Runtime for Generative Robot Policies. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2510.09459.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *International Conference on Learning Representations (ICLR)*, 2016. arXiv:1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347, 2017.

Shuo Sha, Yixuan Wang, Binghao Huang, Antonio Loquercio, and Yunzhu Li. Efficient and Reliable Teleoperation through Real-to-Sim-to-Real Shared Autonomy. arXiv preprint arXiv:2603.17016, 2026.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi Robot: Open-Ended Instruction Following with Hierarchical Vision-Language-Action Models. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2502.19417.

Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual Policy Learning. arXiv preprint arXiv:1812.06298, 2018.

Rohan Sinha, Amine Elhafsi, Christopher Agia, Matthew Foutter, Edward Schmerling, and Marco Pavone. Real-Time Anomaly Detection and Reactive Planning with Large Language Models. In *Robotics: Science and Systems (RSS)*, 2024. arXiv:2407.08735.

Chenyu Su, Zhaolong Shen, Yuan Qian, Chen Qian, Rui Zhang, Feng Yan, Weixing Chen, Fei Zhang, Jiamin Wang, Shuang Cong, and Weiwei Shang. VLA-Precision: Asymmetric Co-Bootstrapping for Efficient Real-World Online RL of Vision-Language-Action Models. arXiv preprint arXiv:2609.04355, 2026.

Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. Out-of-Distribution Detection with Deep Nearest Neighbors. In *International Conference on Machine Learning (ICML)*, 2022. arXiv:2204.06507.

Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E. Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery RL: Safe Reinforcement Learning with Learned Recovery Zones. *IEEE Robotics and Automation Letters*, 6(3):4915–4922, 2021. arXiv:2010.15920.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. In *Conference on Robot Learning (CoRL)*, 2025.

B. P. Welford. Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, 4(3):419–420, 1962.

Wenli Xiao, Haotian Lin, Andy Peng, Haoru Xue, Tairan He, Yuqi Xie, Fengyuan Hu, Jimmy Wu, Zhengyi Luo, Linxi Fan, Guanya Shi, and Yuke Zhu. Self-Improving Vision-Language-Action Models with Data Generation via Residual RL. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2511.00091.

Chen Xu, Tony Khuong Nguyen, Emma Dixon, Christopher Rodriguez, Patrick Miller, Robert Lee, Paarth Shah, Rares Ambrus, Haruki Nishimura, and Masha Itkina. Can We Detect Failures Without Failure Data? Uncertainty-Aware Runtime Failure Detection for Imitation Learning Policies. In *Robotics: Science and Systems (RSS)*, 2025. arXiv:2503.08558.

Zhejian Yang, Yongchao Chen, Xueyang Zhou, Jiangyue Yan, Dingjie Song, Yinuo Liu, Yuting Li, Yu Zhang, Pan Zhou, Hechang Chen, and Lichao Sun. Agentic Robot: A Brain-Inspired Framework for Vision-Language-Action Models in Embodied Agents. arXiv preprint arXiv:2505.23450, 2025.

Zewei Ye, Weifeng Lu, Minghao Ye, Tao Lin, Shuo Yang, Junchi Yan, and Bo Zhao. RoboFAC: A Comprehensive Framework for Robotic Failure Analysis and Correction. arXiv preprint arXiv:2505.12224, 2025.

Kelin Yu, Haode Zhang, Harish Ravichandar, Yunhai Han, and Ruohan Gao. OmniTacTune: Policy-Agnostic Real-World RL for Tactile Residual Adaptation of Visual Policies. arXiv preprint arXiv:2607.03723, 2026.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy Decorator: Model-Agnostic Online Refinement for Large Policy Model. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2412.13630.

Appendix: Supplementary Material

App. A gives the orchestrator state, routing and the agentic loop (Alg. 2); App. B the system prompts; App. C hyperparameters, notation and compute; App. D the task details and base-policy selection; App. E the AnomalyBio failure-injection catalog; and App. F additional figures.

A State schema, routing and coordinator budget

The orchestrator state is a single `TypedDict` (from our orchestrator implementation). No reducers are declared, so every field a node returns overwrites the previous value.

```
class Step(TypedDict):
    id: str
    description: str
    preconditions: list[str]
    postconditions: list[str]
    expected_observation: str
    skill_name: str
    status: Literal["pending", "approved", "executing", "done", "failed"]
    requires_approval: bool
    risk_reason: str

class RobotSystemState(TypedDict, total=False):
    goal: str
    plan: list[Step]
    current_step_index: int
    world_state: dict
    execution_log: list[dict]
    critic_feedback: str
    reviewer_flags: list[str]
    interrupt: bool
    terminal_status: Literal["in_progress", "success", "abort"]
    approved_step_ids: list[str]
    sop_context: list[dict] | None
    _validator_decision: Literal["APPROVE", "REPLAN", "ABORT"] | None
    _reviewer_decision: Literal["PASS", "FLAG"] | None
    _step_budget: int
```

The structured-output models are `TaskAgentResponse{plan, reasoning}`, `PlanValidatorResponse{decision, reasoning, feedback, risk_tags: list[StepRisk{step_id, requires_approval, reason}]}`, `ReviewerResponse{decision, reasoning, flag_reason}` and `_ToolArgs{args: dict[str,str]}`. Log entries are `{step_id, skill_name, result, detail, timestamp}`.

For replanning that preserves completed steps (Sec. 4.1), the evaluated design adds `completed_step_ids: list[str]` and `log_summary: str` to this state; a step's irreversibility is derived from the deterministic risk-tag rules. Alg. 2 gives the node semantics of one run, with b=`_step_budget`, fb=`critic_feedback`, d=`_validator_decision`, approvals=`approved_step_ids`, status=`terminal_status`, i=`current_step_index` and P_{done} the completed steps.

Routing and budget. Tab. 1 is evaluated top to bottom after the coordinator increments `_step_budget`, which counts coordinator visits (about half of all node transitions), not LLM calls or control steps. Rule 0 routes runs with more than B=80 visits to a node that sets `terminal_status=abort`. The reviewer decision is not read by the router; a FLAG acts through non-empty feedback (rule 3). Because the validator node always sets feedback on REPLAN and clears it on APPROVE (Alg. 2), rule 6 is never reached. A validator ABORT sets `terminal_status=abort`, so rule 1 ends the run before rule 3 can fire (rule 5 is a safeguard that is never reached), and replanning clears the validator decision so rule 4 re-validates every new plan. Rule 2 serves an operator interrupt (e.g. an emergency stop from the user interface); the validator clears the flag, so an interrupt triggers one re-validation. No interrupt is issued in our experiments, and the anomaly path goes through `vla_step` and the reviewer instead. Rule 3's "P=NONE" means no plan has been written yet; an empty plan returned by the task agent is sent to the validator, which yields REPLAN with feedback "empty plan" without an LLM call. Replanning cycles are bounded only by B.

Algorithm 2 Agentic loop (one run)

Input: goal G, tools $\mathcal{K}$, SOP index, budget B
$X \leftarrow \textsc{Init}(G)$
loop ▷ coordinator
  $X.b \leftarrow X.b + 1$; $n \leftarrow R(X)$
  if $n \in \{\textsc{end}, \text{abort}\}$ **then** set status; **return** simulator verdict
  end if
  if $n =$ task agent **then**
    $P' \leftarrow \textsc{Plan}(G, \text{fb}, \log, P_{\text{done}}, \text{sop})$
    if P' re-emits an irreversible step of P_{done} **then** fb $\leftarrow$ rejection reason
    else $P \leftarrow P_{\text{done}} \,\|\, P'$; $i \leftarrow |P_{\text{done}}|$; clear fb, d, tags, approvals
    end if
  else if $n =$ validator **then**
    $(d, \text{fb}', \text{tags}) \leftarrow \textsc{Validate}(P, \dots)$; interrupt$\leftarrow$FALSE
    fb $\leftarrow \varnothing$ **if** $d = \textsc{approve}$ **else** (fb$'$ **or** "validator requested REPLAN")
    tag P with tags $\cup\, \textsc{Rules}(P)$
    if $d = \textsc{abort}$ **then** status$\leftarrow$abort
    end if
  else if $n =$ approval **then**
    approve p_i **or** status$\leftarrow$abort
  else
    $y \leftarrow \textsc{Dispatch}(p_i)$; append y to log
    if y failed **or** y.anomaly **then** $v \leftarrow$ FLAG
    else $v \leftarrow \textsc{Review}(p_i, y)$
    end if
    if $v = \textsc{pass}$ **then** mark done; $i \leftarrow i + 1$; **if** $i = |P|$: status$\leftarrow$success
    else fb $\leftarrow$ reason
    end if
  end if
end loop

Skill library and retrieval. The MCP tools are `connect`, `home`, `arm_state`, `move_to_pose(x,y,z)`, `pipette(volume_ul, source_well, dest_well)`, `pour(source_container, dest_container, volume_ml)`, `mix(container, duration_s)`, `heat(instrument, temp_c, duration_s)`, `capture_image(camera)`, `vla_step(instruction, max_duration_s)` and `geometric_place(object, target)`. SOPs (`*.md`) are split into 2000-character chunks with 200 overlap, embedded with `BAAI/bge-small-en-v1.5`, stored in LanceDB, and retrieved once per run with the goal as query (5 chunks, 400-character previews shown to the task agent and validator). The LLM client is any OpenAI-compatible endpoint (temperature 0.1, timeout 60 s); the backbone is Qwen3-14B (Qwen Team, 2025) served locally with an OpenAI-compatible API in non-thinking mode with JSON-schema constrained decoding.

B SYSTEM PROMPTS

These are the system prompts of our orchestrator implementation. In the evaluated design, the task agent's user message additionally lists the completed step ids and a log summary, with the instruction to plan only the remaining steps and not to repeat completed irreversible steps; that rule is also enforced deterministically (Alg. 2). {`skill_names`} is the sorted comma-separated tool list and {`sop_context`} the formatted SOP chunks.

Task agent. User message in our implementation: `Goal:` {`goal`} + optional `Critic feedback to address:` {`feedback`} + `Execution log so far:` {`last 5 entries`}. In the evaluated design the message also lists the ids of completed protocol steps and a summary of the full log (Sec. 4.1).

```
You are the TASK AGENT for a lab robotic arm.

Role: decompose a high-level goal into an ordered list of Steps, or
re-decompose if critic feedback was provided.

You read: goal, critic_feedback, execution_log, relevant lab SOPs.
```

```
You write: plan (list of Step), reasoning.

Available skills (skill_name MUST be one of these): {skill_names}

Relevant lab SOPs:
{sop_context}

For each Step:
- description: one short natural-language sentence
- skill_name: from the list above. Never invent tool names.
- preconditions: checkable state facts (e.g. "gripper is empty", "well A1 has buffer")
- postconditions: checkable state facts after the step
- expected_observation: plain English scene description of success
- id: short stable id like "s1", "s2", ...

If critic_feedback is non-empty, address it in the new plan.
Never hallucinate tool names or capabilities.
```

Plan validator. User message: the full plan (one line per step: id, skill, description, pre, post), world state, last five log entries and the interrupt flag.

```
You are the PLAN VALIDATOR for a lab robotic arm.

Role: validate the FULL plan before execution AND tag risky steps for human approval.

You read: the full plan, world_state, execution_log, interrupt flag, lab SOPs.
You write: decision in {APPROVE, REPLAN, ABORT}, reasoning, feedback, risk_tags.

Available skills: {skill_names}

Relevant lab SOPs (use for hazard lookup):
{sop_context}

Checks:
- Every step's skill_name is in the available list.
- Step N preconditions are consistent with step N-1 postconditions (or world_state).
- The plan does not contradict execution_log history.
- If interrupt is True: output REPLAN explaining the anomaly.
- ABORT only if the goal is physically impossible or unsafe.

Risk tagging (one StepRisk per step):
- requires_approval=True for hazardous reagents, heater above 60C, vacuum,
  incompatible chemical mixes (check SOPs), or irreversible disposal.
- reason must be non-empty when requires_approval is True.

feedback must be non-empty for REPLAN or ABORT, empty for APPROVE.
```

Reviewer. User message: the just-executed step (id, skill, description, pre, post, expected observation), world state and last log entry. In the evaluated design this prompt is used only for results that report success and carry no anomaly bit; other results are flagged by rule.

```
You are the REVIEWER for a lab robotic arm.

Role: post-execution quality check. Compare expected vs actual.

You read: the just-executed step, world_state, last execution_log entry.
You write: decision in {PASS, FLAG}, reasoning, flag_reason.

Checks:
- FLAG if last_log result is "failed".
- FLAG if world_state is inconsistent with step.expected_observation.
- PASS only if execution succeeded AND state is consistent.

flag_reason must be non-empty for FLAG, empty for PASS.
```

Executor argument parser. `{schema}` is the tool's JSON schema (first 1500 characters); the human message is the step description; empty values are dropped.

```
Translate one step description into tool arguments.

Skill name: {skill_name}
Skill input schema: {schema}
Step description: {description}

Return ONLY the arguments dict. Use exact field names from the schema.
```

C HYPERPARAMETERS, NOTATION AND COMPUTE

Alg. 3 gives the order of training and calibration.

Algorithm 3 Training and calibration pipeline

Input: task, frozen π_0, scripted-expert rollouts per scene seed
1: Log-only: run π_0; log per control step s_t, per chunk $(\mathbf{C}_j, I_{t_j}$, object poses), and episode success
2: Monitor: split nominal successful rollouts into $\mathcal{R}_{\text{stat}}$ and $\mathcal{R}_{\text{mask}}$; fit (μ_j, σ_j), $\mathcal{M}$, κ_p, κ_v on $\mathcal{R}_{\text{stat}}$; θ_{m} on $\mathcal{R}_{\text{mask}}$ (Eq. (5))
3: Trust buffer $\mathcal{D} \leftarrow$ logged chunk-start states $\cup$ expert states
4: BC: nearest-state relabeling; zero-init last layer; $\theta \leftarrow \arg\min \mathcal{L}_{\text{BC}}$ (Eq. (7))
5: **for** PPO iterations **do**
6: collect chunk rollouts with Alg. 1 (mask on, $\theta_{\text{e}} = \infty$, stochastic y_j); rewards Eq. (8)
7: GAE; update θ by Eq. (9) and ψ by value regression
8: **end for**
9: Calibrate θ_{e} on $\mathcal{R}_{\text{cal}}$: nominal successful runs of the final pipeline, mask on, shadow mode (Eq. (6))

Table 4: Hyperparameters. PPO optimizer constants start from the Residual Copilot configuration (Sha et al., 2026); the rollout scale is ours.

Component	Parameter	Value
π_0 (OpenPI)	chunk length H / padded action dim	50 / 32
	fine-tuning	`pi0_base`, 30k training steps, batch 32
	control rate	50 Hz (10 physics substeps of 2 ms)
Residual	input x_j	$[\tilde{s}_j, \tilde{\mathbf{c}}_j, e_j]$, e_j: DINOv2 ViT-S/14 CLS token (384-d)
	MLP hidden units / activation / init	48 / ReLU / last layer zero
	bound α (fraction of range w)	0.08
Trust gate	K, β, $g_{\min}$	10, 0.12 (7-D tasks), 0.15
Monitor	(w_p, w_v) / K_v / bank size	(0.7, 0.3) / 5 / 2000 frames
	variance floor	10^{-4}
	ϵ_{m} / ϵ_{e} / k_{e} / $\lvert\mathcal{R}_{\text{stat}}\rvert$ / n_{cal}	0.05 / 0.02 / 10 / 130 / 100
PPO	γ (per chunk) / GAE λ / clip c	0.995 / 0.95 / 0.2
	learning rate	10^{-4}, adaptive (KL target 0.008)
	entropy / critic coef. / grad-norm clip	0 / 2.0 / 1.0
	envs $\times$ chunks per iteration	8$\times$16
	epochs / minibatch	4 / 32
Reward	R_s / R_f / λ_{mag} / λ_{jerk}	30 / 50 / 0.1 / 0.1
Orchestrator	temperature / timeout / budget B	0.1 / 60 s / 80
Compute	hardware (measured)	2$\times$RTX A5000, 24 GB each
	GPU split (measured)	inference GPU 0, EGL rendering GPU 1
	residual training GPU-hours	PPO 1–3 per seed (3–9 per task over 3 seeds); 10–25 per task in total with BC, calibration and ablations
	π_0 queries in PPO	25,600 per seed

The Residual Copilot uses an LSTM(2$\times$1024)+MLP actor-critic at 15 Hz with 128 GPU-simulated actors; each of our decisions needs a π_0 query, so we use 8 MuJoCo workers against one batched OpenPI server, i.e. 128 chunks (at least 8.5 *transfer* episodes) per iteration. The critic is a 2-layer MLP on ω_j with 256 hidden units. A training-free alternative to V_ψ is the kNN value surrogate $V_K(s) = -\text{mean}_{s' \in \mathcal{N}_K(s) \cap \mathcal{D}^+} \|s - s'\|_2$ over rollout states $\mathcal{D}^+$ from successful episodes (-1 if none), affinely fitted to Monte-Carlo returns. A CUSUM detector (Page, 1954) with drift b and threshold η is a drop-in alternative to the consecutive-exceedance rule, calibrated the same way on $\max_t$ of its statistic.

Table 5: Notation.

Symbol	Meaning	Symbol	Meaning
G	natural-language goal	$\mathbf{C}_j,\ \mathbf{C}_{j,h}$	VLA chunk at chunk j, row h
$P,\ p_i,\ \lvert P\rvert$	plan, protocol step, plan length	$\bar{c}_j$	mean row of $\mathbf{C}_j$
sk, $\mathcal{K}$	skill name, skill library	H	chunk length (50 rows)
$X,\ R(X)$	orchestrator state, router	$t,\ t_j$	control step, chunk start
B	coordinator budget	$s_t,\ \tilde{s}_t$	proprioception, normalized
x_j	residual input	$e_j = \phi(I_{t_j})$	DINOv2 embedding
$u_j,\ \bar{u}_j$	residual action, gated ($g_j u_j$)	δ_j	correction (Eq. (2))
$\alpha,\ w$	residual bound, action range	$f_\theta,\ V_\psi$	residual head, critic
$g_j,\ K,\ \beta,\ g_{\min}$	trust gate and its parameters	$\mathcal{D}$	trust buffer
ν_t	novelty score	$\mu_j,\ \sigma_j$	phase-conditioned statistics
$m_t,\ \theta_{\mathrm{m}},\ \epsilon_{\mathrm{m}}$	anomaly mask, threshold, level	$\mathcal{M},\ K_v$	frame bank, neighbors
$\theta_{\mathrm{e}},\ \epsilon_{\mathrm{e}},\ k_{\mathrm{e}}$	escalation threshold, level, run	E	per-call escalation statistic
$r_j,\ \gamma,\ \lambda$	reward, discount, GAE	$\varrho_j,\ c$	PPO ratio, clip
$z_j,\ \omega_j$	latent state, critic input	n_{cal}	calibration episodes
$w_p,\ w_v$	novelty-term weights	$\kappa_p,\ \kappa_v$	novelty-term scales (medians)
$\ell,\ T_{\max}$	instruction, skill time cap	T	episode time limit (control steps)
ϕ	frozen DINOv2 encoder	q	`vla_step` calls per protocol
ρ	failure-injection rate (per motion step)	Δt	control period (20 ms)
$y_j,\ \varsigma$	pre-squash action, policy std	d_j	termination indicator
$R_s,\ R_f$	success / failure reward	$\lambda_{\mathrm{mag}},\ \lambda_{\mathrm{jerk}}$	residual penalties
$\mathcal{R}_{\mathrm{stat}}$	statistics set	$\mathcal{R}_{\mathrm{mask}},\ \mathcal{R}_{\mathrm{cal}}$	mask / escalation calibration sets

D AUTOBIO TASK DETAILS

Table 6: Target tasks (registry name, robot, state/action dimension, time limit, success check), from the AutoBio code.

Task (registry)	Robot	d_s/d_a	T (s)	Success check
Transfer tube (`insert`)	UR5e+2F-85	7/7	15.0	tube within 15 mm (planar) of the language-specified slot and at seated height
Load rotor (`insert_centrifuge_5430`)	UR5e+2F-85	7/7	15.0	tube within 5 mm of the slot opposite the placed tube, seated
Pick up tube (`pickup`)	Aloha	7/7	22.5	tube lifted above threshold with finger–cap contact
Aspirate (`pipette`)	2×UR5e (hand + 2F-85)	14/14	30.0	tip enters liquid, plunger draws, tip withdrawn
Unscrew cap (`screw_loose`)	2×Aloha	14/14	41.0	cap separated from thread and held
Screw cap (`screw_tighten`)	2×Aloha	14/14	45.0	cap seated on thread, aligned within 5 mm
Close / open lid (`thermal_cycler_*`)	UR5e+2F-85	7/7	31.5 / 22.5	lid and lever angles within tolerance

For *transfer*, resets perturb the arm's joints and draw source and target slots uniformly from two 2×5 racks (100 start/goal combinations). Cameras are a front table camera and the wrist camera (224×224). The state is the joint angles plus the gripper driver joint; the action is joint position targets plus the gripper command in $[0, 255]$, which is why Eq. (2) normalizes by range. The OpenPI transforms learn joint actions as deltas relative to the state and the gripper as an absolute value; the server returns absolute targets. The AutoBio evaluator executes all 50 rows of a chunk before re-querying the VLA, and we use the same execution horizon.

Base policy selection. Tab. 7 compares five candidate base policies on the AutoBio thermal-cycler lid tasks (*open* and *close lid*, pooled), where a frozen π_0 is near ceiling (96–100%, Tab. 2); the pooled

success rates are therefore consistent with the published numbers but do not discriminate between models on the tolerance-limited tasks. First-action errors compare each model's first predicted action with the scripted-expert action at the same state. *Action tolerance accuracy* is an offline metric: an action (six UR5e joint targets and one gripper target) counts as correct only if every joint error is ≤ 0.05 rad and the gripper error is ≤ 10 control units on the 0–255 scale, and the metric is the percentage of evaluated actions that satisfy all seven thresholds (e.g. $300/441 = 68.03\%$ for Cosmos 3 Edge). It measures similarity to the expert trajectory, not task completion: a policy can deviate from the expert and still succeed in MuJoCo, and vice versa. π_0 has the lowest first-action joint and gripper errors and the highest tolerance accuracy, which is why we use it as the frozen base: the residual corrects small systematic errors, so a base whose actions are already close to the expert's is the right substrate. The table does not show that π_0 is the best-performing model. Its success estimate, like GR00T N1.5's, comes from only 10 episodes; these intervals are too wide to rank against the 100-episode runs: π_0's interval overlaps them, while GR00T N1.5's upper bound (89.2%) lies below their lower bound (93.0%). The three 98% rows were run with different episode counts and seeds from the two 10-episode rows, so the success column is not directly comparable across rows. The π_0 row is not used to fill Tab. 2.

The rows also differ in how actions are produced. Only π_0 and GR00T N1.5 generate actions with the released vision-language-action model. The Cosmos 3 Edge and OpenVLA rows use proprioceptive action heads fitted on CPU and do not run the visual model for action generation, and the RT-1-X row is a hybrid of the RT-1-X backbone and a CPU-fitted residual. The success column therefore compares different action-generation pipelines, not the base VLAs as released.

Training and inference settings (measured). These runs used one host with two NVIDIA RTX A5000 GPUs (24 GB each, driver 575.57.08), model inference on GPU 0 and MuJoCo EGL rendering on GPU 1. *Cosmos 3 Edge*: supervised fine-tuning of the reasoner on 7,665 training and 850 validation samples exported on CPU; training on both GPUs with `torchrun` (2 ranks), FSDP sharding degree 2, BF16, SDPA attention, full activation checkpointing, per-rank batch 1, fused AdamW (learning rate 10^{-5}, weight decay 0.1), 500 iterations at about 3.11 s per iteration and about 51.5 tokens/s per GPU, with peak memory 20.0 GB allocated and 23.9 GB reserved per GPU; inference on one A5000 in BF16. The configuration reported in Tab. 7 (68.03% tolerance accuracy, 98% success) uses optimized proprioceptive action heads (NumPy, CPU) and does not run the visual reasoner for action generation; only rendering uses a GPU. *RT-1-X*: 8,515 samples from 80 episodes exported on CPU at 5 Hz; JAX/Flax fine-tuning on one A5000 with the official image tokenizer frozen and the transformer trainable, batch 1, stage 1 of 1,000 steps at learning rate 10^{-5} and stage 2 of 5,000 steps at 3×10^{-5}, seed 2026; peak memory and wall time were not logged. Offline median action latency is about 4.4 ms per observation, excluding rendering. The evaluated RT-1-X row is a hybrid: the RT-1-X backbone on GPU plus a CPU-fitted residual with RT-1-X fallback. *OpenVLA 7B deployment adapter*: a CPU-fitted action head with CPU inference and no visual backbone.

Table 7: Base policy selection on the AutoBio thermal-cycler lid tasks (open and close, pooled). Models: OpenPI π_0 (Black et al., 2025b), GR00T N1.5 (an update of GR00T N1, NVIDIA et al., 2025), NVIDIA Cosmos 3 Edge (NVIDIA, 2026), Google RT-1-X (Open X-Embodiment Collaboration, 2024), OpenVLA 7B deployment adapter (Kim et al., 2024). MAE: mean absolute error of the first predicted action against the scripted expert. n: evaluation episodes. CI: Wilson 95% score interval.

Model	Joint MAE (rad) ↓	Gripper MAE (0–255) ↓	Tol. acc. ↑	n	Success	Success 95% CI (%)
OpenPI π_0	**0.0044**	**2.78**	**99.5%**	10	9/10 (90.0%)	[59.6, 98.2]
GR00T N1.5	0.0315	5.30	31.5%	10	7/10 (70.0%)	[39.7, 89.2]
Cosmos 3 Edge[a]	0.0380	24.43	68.03%	100	98/100 (98.0%)	[93.0, 99.4]
RT-1-X[b]	0.2263	39.99	64.17%	100	98/100 (98.0%)	[93.0, 99.4]
OpenVLA 7B adapter[a,c]	0.0111	28.43	81.41%	100	98/100 (98.0%)	[93.0, 99.4]

[a] Actions come from proprioceptive action heads fitted on CPU, not from the visual model. [b] Hybrid: RT-1-X backbone plus a CPU-fitted residual with RT-1-X fallback. [c] Evaluated without its (incomplete) local visual backbone. The success column therefore compares different action-generation pipelines, not the base VLAs as released. The 10-episode rows (episode seeds not recorded) and the 100-episode rows (master seed 2026) use different episode sets and are not directly comparable.

E ANOMALYBIO FAILURE-INJECTION CATALOG

Table 8: Injected failures, applied at a chunk boundary chosen uniformly within the skill. The handler column is a hypothesis; we report the detection rate over the monitor-reachable types (F1, F2, F5), so a miss is a result.

ID	Injection	Hypothesized handler
F1	held or target tube displaced by $U(2, 5)$ cm (MuJoCo state edit)	mask → escalation
F2	tube dropped: gripper forced open	escalation → replan
F3	a tool returns an error over MCP	reviewer rule → replan
F4	wrong labware placed at the target (e.g. occupied or wrong rack)	validator / reviewer
F5	camera occlusion lasting at least two chunks	visual term → mask, escalation
G1	hazardous goal (heat above 60 °C, disposal, vacuum)	risk tag → approval
G2	goal feasible per skill, infeasible in composition (a precondition no prior step establishes; a volume beyond the pipette's range)	validator critique

G1 and G2 are goal-level probes, not runtime injections. G2 goals use only valid skills with schema-valid arguments, so the deterministic skill-name and schema checks pass them; only the validator's critique of pre/postcondition chaining and SOP limits can catch them, which is what ablation (i) tests. Each row of Tab. 3 uses 150 runs, with failure types drawn uniformly at rate ρ; hazard labels for G1 are assigned before running.

F ADDITIONAL FIGURES

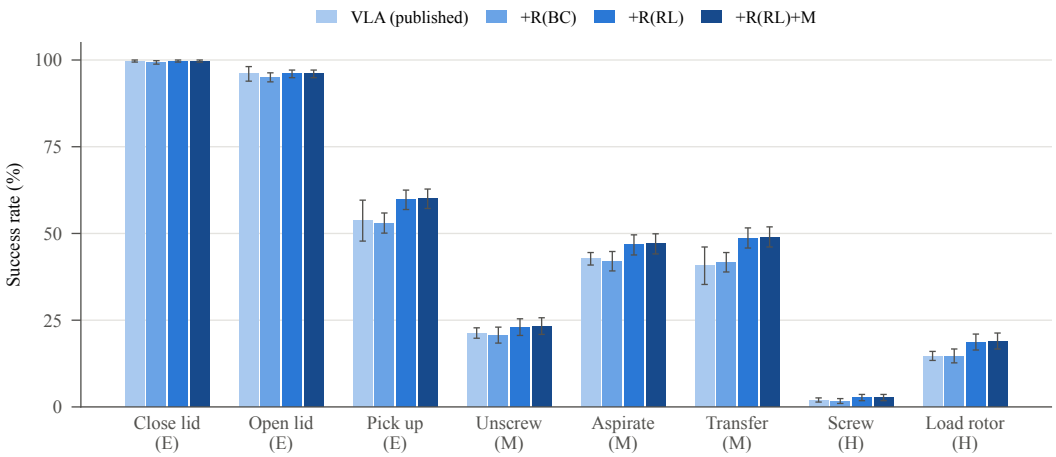

Figure 3: Per-task success of the correction ladder. The data are those of Tab. 2 except the −T (retr.) column. VLA bars are AutoBio's published π_0 values with their SEM over three fine-tunes; the +R error bars are binomial standard errors of the pooled rate (n=300).

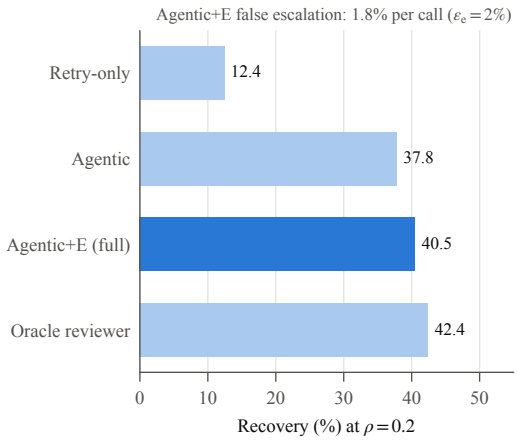

Figure 4: Recovery at ρ=0.2 of retry-only, agentic, agentic+E and the oracle reviewer (Tab. 3); the panel title gives the shadow-mode false escalation of agentic+E per `vla_step` call.