CERES: CAPABILITY-AWARE FAST-SLOW UNDERWATER NAVIGATION WITH CORRECTIVE EXPERTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Robots operating with heterogeneous sensing face two coupled deployment decisions: whether a responsive default policy remains inside its capability envelope and, when correction is valuable, which control variable additional computation should revise. Degraded representations, unsafe current actions, and persistent plan failures demand distinct authority and effect times. We present *CERES* (Capability-Envelope Routing with Expert Specialization), a fast–slow controller that turns these decisions into cost-aware selection among causally typed interventions. System 1 continuously provides a bounded proposal and competence evidence; a coordinator combines learned correction value, realized-outcome memory, and an admissibility contract to preserve fast control or invoke System 2. System 2 centers on two current-decision experts: a State Refiner repairs the representation and a Safety Verifier certifies or projects the action; a Capability Planner conditionally extends correction to subsequent context under verified delayed execution. Across 72 paired trials in three underwater scenes, CERES preserves all 48 routine System 1 successes while recovering 14/24 challenging trials, reaching 86.1% overall success versus 66.7% for System 1 and 72.2% for always-on System 2. It reduces System 1 collision from 16.7% to 1.4% and always-on normalized expert-forward load by 39.5%. At matched success with a near-always-on risk rule, CERES uses 29.6% fewer active experts per decision. Targeted evaluations establish strong representation repair, complementary action-risk correction, verified delayed planning semantics, and a complete fast path below 50 ms at P99.

1 INTRODUCTION

Learned robot control must reconcile rapid action generation with the ability to recover when perception, action selection, or task progress becomes unreliable. Underwater navigation brings these demands together: visual and acoustic observations can degrade differently, limited clearance makes action errors consequential, and locally collision-free behavior can persist without reaching the objective. Multimodal underwater policies have demonstrated navigation under challenging sensing conditions (Yang et al., 2023), while generalist robot policies and vision–language–action (VLA) models have expanded the scope of learned action generation (Ghosh et al., 2024; Kim et al., 2025; Shukor et al., 2025). Deploying such policies in a closed loop also requires deciding when their current proposal remains adequate and how additional computation should intervene when it does not.

Adaptive computation offers a natural starting point. Fast–slow models balance direct prediction and deliberative planning (Su et al., 2025; Saha et al., 2025), input-adaptive allocation estimates where additional inference is most valuable (Damani et al., 2025), and sparse mixture-of-experts (MoE) models route inputs to selected computational modules (Fedus et al., 2022). In robotics, OneTwoVLA integrates reasoning and acting within a unified policy and invokes reasoning at critical moments, including error recovery (Lin et al., 2026). These advances establish the value of selective computation. For closed-loop correction, however, the allocation decision must also account for the failure being addressed, the variable that an intervention may modify, and the stage at which its update can affect execution.

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.

Two coupled requirements arise. First, computation should be allocated according to the expected benefit of a specific correction in the current control context. Uncertainty, risk, and stalled progress provide useful evidence; however, an indication of difficulty alone does not determine which intervention will help or whether its benefit justifies its cost. A fast policy may already produce an adequate action despite uncertain observations, whereas a repeated low-risk action may require a change of objective. The controller therefore needs to estimate corrective value relative to retaining the fast proposal, using both current evidence and consequences observed after previous actions. This makes preserving competent behavior and recovering lost capability parts of the same allocation problem.

Second, correction requires explicit scope and execution order. Safety layers address action correction (Dalal et al., 2018), and runtime-assurance architectures regulate switching between controllers (Phan et al., 2020). The heterogeneous failures considered here additionally call for coordinated interventions at several points in the control loop. Degraded evidence may require repairing the representation before recomputing a proposal; immediate motion risk may require modifying the current action; persistent objective failure may require revising the context used by subsequent decisions. Selecting a module is insufficient unless its permissible updates and their temporal scope are also specified. In particular, a planning revision must not silently override current-action verification, and an execution outcome can inform correction only after it has been observed.

We address these requirements with CERES (*Capability-Envelope Routing with Expert Specialization*), a fast–slow controller that selects corrective interventions with explicit write targets and effect times. Its coordinator retains the System 1 proposal when correction is predicted to be unnecessary and local admissibility checks are satisfied; these conditions operationally define the fast policy's capability envelope. Otherwise, it selects a legal System 2 configuration. The coordinator learns cost-aware corrective value from matched short-horizon continuations and uses realized-outcome memory to contextualize later decisions. A shared admissibility contract constrains the selected configuration and its execution, connecting learned allocation to the controller's permitted behavior.

System 2 implements three distinct corrective roles. The *State Refiner* (SR) repairs the current representation and triggers a recomputed System 1 proposal. The *Safety Verifier* (SV), when active, verifies or projects the candidate action and has final learned authority over the current command. The *Capability Planner* (CP) conditionally revises subsequent decision context following verified execution. SR and SV provide the principal current-decision corrections, while CP extends the same intervention framework across decisions. Configuration selection thus determines both the corrective mechanism and its predefined scope of influence within the closed loop.

Our contributions are summarized as follows: **(1)** We formulate capability-envelope coordination as cost-aware selection between a fast proposal and legal corrective configurations. The coordinator combines contextual evidence and realized-outcome memory with supervision from matched continuations, learning where mechanism-specific correction offers value under a shared admissibility contract. **(2)** We develop a mechanism-specialized System 2 with explicit write authority and temporal scope. Representation repair, current-action verification, and subsequent-context revision have distinct interfaces and constrained composition, allowing each intervention's effects to be examined at the variable and decision stage it is designed to modify.

We evaluate CERES through paired system controls, matched router comparisons, and mechanism-targeted studies in underwater simulation. Across 72 paired trials, it preserves all 48 routine System 1 successes and completes 14 of 24 challenging trials, achieving 86.1% overall success versus 66.7% for System 1 and 72.2% for always-on System 2, with 39.5% lower normalized expert-forward load than the latter. Router controls show reduced computation at matched observed success, while targeted studies examine representation repair, action correction, delayed planning, and outcome memory separately. Adapted VLA policies provide complementary whole-policy references under the common task and action protocol.

2 Related Work

Underwater and generalist control. Multimodal underwater policies combine visual, acoustic, and goal evidence for collision avoidance (Yang et al., 2023). Generalist robot policies learn transferable action generation from scalable data and pretrained representations (Brohan et al., 2023;

Zitkovich et al., 2023; Open X-Embodiment Collaboration, 2024; Ghosh et al., 2024; Kim et al., 2025; Shukor et al., 2025; Qu et al., 2025; Zheng et al., 2026; Black et al., 2025b;a); recent work studies how VLM competence transfers to action policies (Zhang et al., 2026) and introduces adaptive reasoning within VLA control (Lin et al., 2026). These advances strengthen the policy that proposes an action. CERES addresses the complementary deployment layer around that proposal: preserving competent execution, detecting mechanism-specific corrective value, and granting bounded authority to the corresponding intervention.

Adaptive computation and control hierarchies. Sparse MoE and dynamic-depth models select which or how much computation to activate (Fedus et al., 2022; Mathur et al., 2025); input-adaptive allocation and fast–slow planning route harder inputs toward expensive inference (Damani et al., 2025; Su et al., 2025; Saha et al., 2025). Hierarchical controllers combine reactive execution with slower planning (Hansel et al., 2023). These approaches make computation amount, expert identity, or temporal hierarchy the routing object. CERES makes the corrective intervention itself the routing object: selection jointly specifies resource use, legal write target, and effect time, with realized consequences informing later allocation.

Expert and safety mechanisms. Expert routing distinguishes subnetworks by input region or capacity, while safety layers and action-constrained policies enforce feasible actions (Fedus et al., 2022; Dalal et al., 2018; Hung et al., 2025). CERES unifies these concerns through legal intervention configurations and SR–SV–CP ordering. Representation, current action, and subsequent context receive distinct write authority; SV closes every active current-action pathway, and CP revisions become visible on the next decision. This typed contract turns expert specialization into an executable causal property, which matched routing and mechanism controls test directly.

3 METHOD

3.1 PROBLEM FORMULATION

At decision t, the robot receives onboard RGB, sonar, point, and proprioceptive observations o_t, task specification x^τ, and local context c_t (current goal, observed occupancy and range evidence, and validity masks). It produces a bounded body-frame command $a_t = [v_t, \omega_t]$. A task capability graph $G^\tau = (\mathcal{K}, \mathcal{E}^\tau)$ constrains legal transitions among APPROACH, AVOIDRISK, ALIGN, STABILIZE, RECOVER, and RETREAT. Decisions use the execution-available history $\mathcal{H}_t = \{o_{0:t}, x^\tau, c_{0:t}, a_{0:t-1}, \rho_{1:t}\}$, where ρ contains already observed execution consequences.

Let e_t be causal evidence, $\zeta_t = \emptyset$ retain the fast proposal, and $\zeta_t \in \mathcal{C}_{\text{legal}}(\mathcal{H}_t)$ select a legal corrective configuration. One closed-loop decision is

$$\begin{aligned} e_t = E(\mathcal{H}_t), \qquad \zeta_t = \pi_\theta(e_t) \in \{\emptyset\} \cup \mathcal{C}_{\text{legal}}(\mathcal{H}_t), \\ (a_t, c_{t+1}) = \mathcal{I}_{\zeta_t}(h_t, a_t^{(1)}, c_t), \\ \mathcal{H}_{t+1} = U(\mathcal{H}_t, o_{t+1}, a_t, \rho_{t+1}) \end{aligned} \tag{1}$$

where $\mathcal{I}_{\zeta_t}$ is typed: SR writes the current representation, SV the current action, and CP the next-decision context. The realized ρ_{t+1} becomes available after acting, so each update enters precisely at its declared temporal scope.

3.2 CERES OVERVIEW

Figure 1 instantiates Eq. 1 as *evidence–decision–intervention*: whether correction is warranted, which variable it may change, and when the change takes effect. Multimodal state h_t, System 1 uncertainty u_t, perceptual risk r_t, action inconsistency d_t, failure evidence f_t, capability k_t, risk trend M_t, and realized-outcome memory m_t^{mem} form e_t. Inside the capability envelope, the bounded System 1 proposal $a_t^{(1)}$ retains control; outside it, the coordinator selects from $\mathcal{C} = \{\text{SR}, \text{SV}, \text{SR+SV}, \text{SV+CP}, \text{SR+SV+CP}\}$. Each selected configuration is an executable causal contract that links observed failure evidence to a write target and effect time.

CP is paired with SV, coupling every revised objective to action verification. Selected experts execute in SR–SV–CP order: SR may change the representation and trigger a recomputed fast proposal;

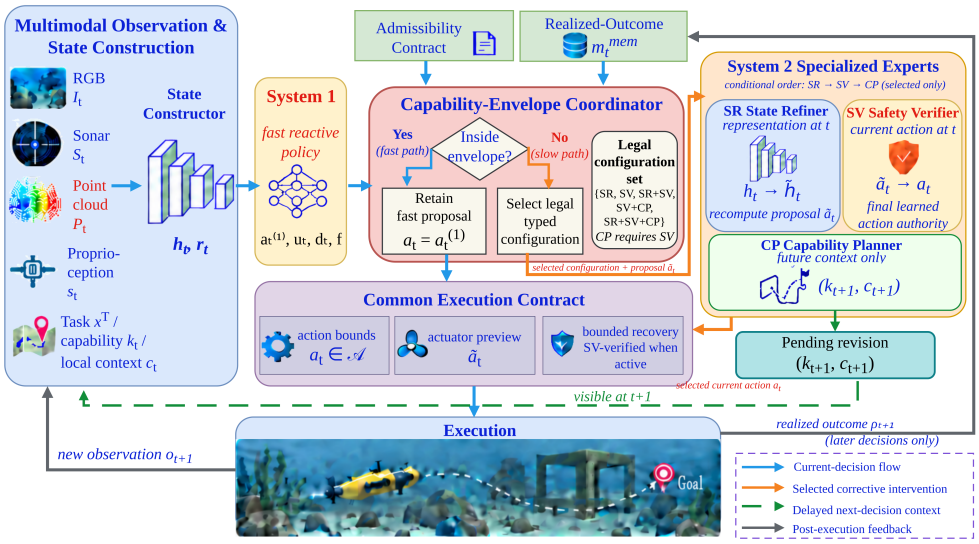

Figure 1: CERES closed-loop architecture. Multimodal observations feed the reactive System 1 policy. The capability-envelope coordinator retains its bounded proposal inside the envelope or selects a legal typed intervention outside it. SR may refine the current representation, SV has final learned authority over the current action, and CP may revise the next-decision context. Realized outcomes enter memory after execution and inform subsequent decisions.

SV certifies the current action; CP updates subsequent context. Execution then produces evidence for a later decision. Appendix A gives the full interfaces and legal configurations.

3.3 Multimodal State and Reactive Control

State construction. Encoders fuse RGB, sonar, point observations, proprioception, and task tokens into h_t; the frozen constructor also supplies the clean reference used to supervise SR. Perceptual evidence r_t summarizes masked modality reliability, cross-modal conflict, capability-conditioned anomaly, and temporal inconsistency, with explicit availability masks for missing measurements. Architectural and training details are in Appendices A–B.

System 1: Reactive Policy. System 1 conditions a short history of h_t on masked occupancy, range, and goal-relative context. A bounded action head and evidential regression head (Amini et al., 2020) produce $a_t^{(1)}$ and epistemic uncertainty u_t; separate diagnostics measure action inconsistency d_t and observed failure evidence f_t from progress, objective deviation, and danger proximity. These signals provide routing and expert evidence alongside the action proposal.

3.4 Capability-Envelope Routing

History-consistent routing state. An exponentially smoothed trace of conflict, action inconsistency, and failure evidence, together with its first difference, distinguishes transient from persistent risk. Capability-indexed memory stores executed decisions after their outcomes arrive; retrieved entries summarize realized return, retrospective benefit, and executed configuration. This realized-outcome feedback aligns every memory item with an action that actually occurred. A learned gate combines a representation branch $(h_t, k_t, m_t^{\mathrm{mem}})$ with a diagnostic branch $(r_t, u_t, d_t, f_t, M_t)$ to predict fast–slow probability p_t^{slow} and configuration probability p_t^{cfg}.

Capability envelope and admissibility contract. Let τ_z be the routing threshold and let χ_t denote a local-admissibility test that checks the bounded proposal against currently observed corridor geometry, actuator stopping distance, target validity, and recent progress. We operationally define

the System 1 capability envelope as the region in which the fast proposal is predicted sufficient for direct execution and locally admissible:

$$\mathcal{E}_\theta = \{\mathcal{H}_t : p_t^{\text{slow}} < \tau_z \ \wedge \ \chi_t(a_t^{(1)}) = 1\}. \tag{2}$$

Inside $\mathcal{E}_\theta$, CERES executes System 1. Outside it, the coordinator proposes a typed configuration and a shared temporally valid admissibility contract projects that proposal onto the legal set. The learned signal estimates whether correction is valuable; contextual selection specifies the candidate write authority, while the contract provides invariant information timing, capability transitions, bounded actions, and verification across all routers.

Cost-aware routing supervision. The routing target is obtained from matched short-horizon continuations that share the same state, controller history, and disturbance realization across the fast path and all legal configurations. For configuration c, we define

$$J(c) = \mathbf{1}[\text{capability complete}] + \lambda_p \Delta_t(c) - \lambda_{\text{col}} \mathbf{1}[\text{collision}] - \lambda_c \frac{|c|}{3}, \tag{3}$$

where Δ_t is clipped local progress and $|c|$ counts experts. The candidate pool prioritizes collision-free continuations, and near-tied utilities favor smaller configurations. The slow target activates when correction rescues an unsafe fast continuation or improves utility by a prescribed margin. Slow-path, configuration, benefit, risk, memory-consistency, and specialization losses train the router (Appendix B), embedding activation cost directly in supervision. Routing thereby estimates mechanism-specific corrective value under explicit cost and admissibility.

3.5 Mechanism-Specialized System 2: Specialized Corrective Experts

State Refiner (SR). SR addresses representation failure. From multimodal evidence, h_t, r_t, and capability context, it predicts a residual $\widehat{h}_t = h_t + \Delta h_t^{\text{SR}}$ and modality confidences. Selecting SR grants latent-write authority, replaces the current latent, and recomputes System 1. An SR-directed trust region bounds the update, and an ensuing SV can verify the recomputed action. SR is supervised by the frozen constructor's clean representation and degradation-derived confidence targets. Its diagnostic forward precedes routing, so we report selection and neural-forward intensity separately.

Safety Verifier (SV). SV addresses current-action risk. From candidate $\widetilde{a}_t$, effective representation, risk and failure evidence, and capability, it predicts a veto, future risk, and $\mathcal{A}_t^{\text{safe}} = \{a : A_t a \leq b_t\} \cap \mathcal{A}$. It can reduce forward velocity before exact projection,

$$a_t^{\text{SV}} = \arg\min_{a \in \mathcal{A}_t^{\text{safe}}} \|a - \widetilde{a}_t\|_2^2. \tag{4}$$

If the set is empty, the robot holds and re-evaluates. Whenever SV is active, Eq. 4 has final learned authority over the current command. Detailed auxiliary supervision appears in Appendix B.

Capability Planner (CP). CP addresses persistent objective failure at the cross-decision timescale. From task, effective state, failure trend, and current capability, it predicts a revision event, graph-masked successor, and local subgoal. After persistence and lifecycle checks, a revision proposed at t is committed as (k_{t+1}, c_{t+1}) and first becomes visible at $t+1$. CP follows a verified motion event, preserving a clean boundary between current-action certification and subsequent-context revision. Its event, successor, and subgoal objectives are detailed in Appendix B.

3.6 Execution-Consistent Coordination and Structured Optimization

CERES constructs the fast proposal, applies selected SR and recomputes it, incorporates previously committed CP context, and finally applies SV; a new CP proposal is registered for the next decision. The learned router proposes whether and how to intervene, the admissibility contract checks the current proposal, and bounded recovery handles infeasible commands using observed map/range evidence. These components jointly determine the executed action; recovery motion is also verified when SV is active.

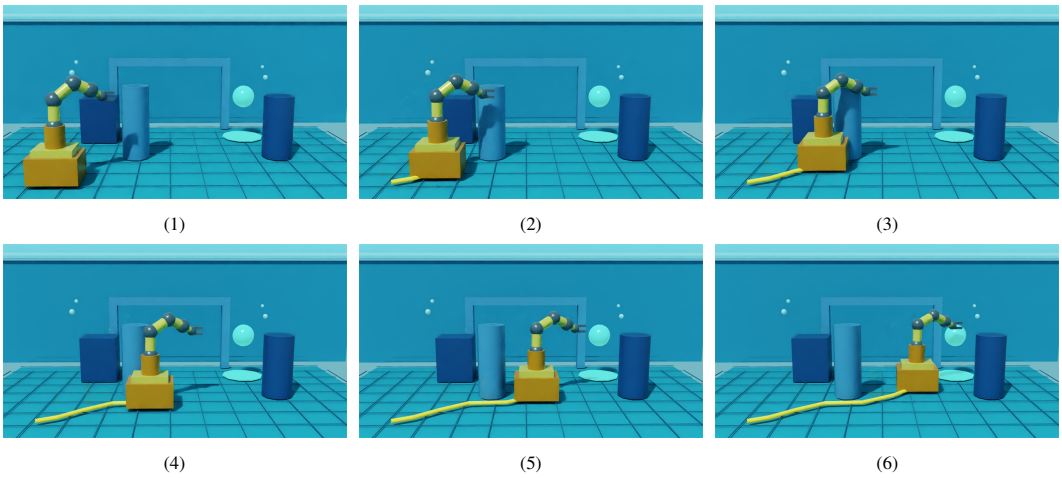

Figure 2: Fixed-view Isaac Sim replay of one successful CERES navigation trial in IndustrialFrame. Frames (1)–(6) show the task execution in chronological order; the yellow curve is the executed trajectory accumulated at each depicted point.

This ordering enforces three invariants: post-execution information begins informing subsequent decisions, CP first affects the decision after its proposal, and expert removal removes the corresponding write semantics. The state constructor and System 1 are frozen before corrective learning; coordinated optimization updates routing and downstream composition while keeping the constructor, System 1, and SR fixed. Freezing preserves expert identity and makes mechanism ablations interpretable. At deployment, capability lifecycle, diagnostics, and realized-outcome memory provide the evolving controller state; Appendix B gives the schedule.

4 Experiments

We test whether governed correction resolves the two practical failures of uniform computation: insufficient capability when System 1 leaves its envelope, and unnecessary intervention when System 1 is already competent. We further test whether representation, action, and future-context authority produce mechanism-specific effects at their declared temporal scopes. Paired System 1/always-on controls, router isolation, expert-specific interventions, and seven VLA backbones in ten adaptation configurations evaluate the controller from complementary system and mechanism perspectives.

4.1 Experimental Setup

Closed-loop protocol. Isaac Sim 4.5 evaluation uses 72 outcome-independent trials (48 routine, 24 challenging) balanced across *IndustrialFrame*, *OpenTank*, and *PipeCorridor*. Paired controllers share task, seed, physical/observation initial state, disturbance schedule, and 960-decision horizon, and every registered trial enters the analysis. A trial succeeds after the vehicle remains within 0.78 m of the goal and 0.08 rad of the target heading while stationary for eight consecutive decisions. Physics advances in 0.05 s increments; policy latency is measured independently. Full task definitions are in Appendix C.

Figure 2 gives a fixed-view chronological visualization of a successful IndustrialFrame trial, with the executed route accumulating through the complete obstacle field.

Training populations. CERES is trained from Train700 (198,721 decision frames), with disjoint Dev100 and registered Paper72 populations. Every external adaptation draws from the same Train700 episode identities; neither Dev100 nor Paper72 outcomes enter training. Frame counts and model-native samples are different optimization units, detailed in Appendix F.

System and VLA baselines. Architectural controls are the frozen fast policy alone and the same system with System 2 always invoked. External references comprise seven VLA backbones in ten

configurations. NTA trains architecture-supported action modules of OpenVLA-7B, SmolVLA, and Octo-Base-1.5 (Kim et al., 2025; Shukor et al., 2025; Ghosh et al., 2024); FLA freezes those and four additional backbones and fits a common two-dimensional action bridge (Qu et al., 2025; Zheng et al., 2026; Black et al., 2025b;a).

NTA takes 16 deterministic integer-spaced anchors, including endpoints, from every Train700 episode: 11,200 outcome-independent samples with equal episode weight. Seven hundred steps at effective batch 16 consume this complete population. OpenVLA trains rank-32 QLoRA over its vision projector and language–action policy, SmolVLA its native action expert and state/action/time projections, and Octo a 2-D action head plus supported non-Transformer modules (110.8M, 99.9M, and 7.1M trainable parameters). Disjoint Dev100 anchors select checkpoints, with improved native objectives and nonzero parameter updates. FLA fits its closed-form bridge from four anchors per Train700 episode (2,800 total).

Every VLA receives current RGB, task text, and the same causal body-frame goal; SmolVLA also receives supported proprioception, while CERES retains its complete onboard interface. Evaluation shares the exact 72 task slots, seeds, initial states, disturbances, horizon, action bounds, goal semantics, dynamics, and outcomes. Table 3 compares complete policies under these native causal interfaces; matched internal controls isolate routing and typed intervention.

Router-isolation controls. Four pre-specified controls isolate the routing proposal: a frozen learned fast–slow signal, uncertainty thresholding, fixed risk rules, and a frozen-distribution timing diagnostic. Experts, memory initialization, deterministic admissibility and recovery, environment, random state, and actuation remain identical across all 288 trajectories, attributing compute-allocation differences directly to routing (Appendix D).

Metrics and statistics. We report completion (success; normalized progress P), safety (collision), path efficiency (SPL), and computation (System 2 duty, normalized expert-forward load, and router-call count). Duty and normalized load are computed within each episode and then averaged across episodes; router-isolation tables additionally report total expert forwards. Router outcomes and computation are interpreted jointly. Binary endpoints use exact paired McNemar tests, continuous endpoints use two-sided paired permutation tests, effects use 10,000 paired bootstrap resamples, and Holm correction is applied within pre-specified families. Appendix C defines all metrics and families.

4.2 System 1/System 2 Coordination

Table 1: System-level results on 72 paired trials. Rout. and Chal. are routine- and challenging-task success; C/100m is collision density; Unsafe-E/D are unsafe-episode/decision rates; Head is terminal heading error; Path is traveled-distance ratio. Load is the episode-averaged number of completed SR/SV/CP forwards divided by three times the episode's decision count. Success, collision, unsafe incidence, and episode-averaged System 2 duty are percentages.

A. Completion and safety

Controller	Succ.↑	Rout.↑	Chal.↑	Coll.↓	C/100m↓	Unsafe-E↓	Unsafe-D↓
System 1 only	66.7	**100.0**	0.0	16.7	13.97	18.1	1.76
Always-on S2	72.2	83.3	50.0	**0.0**	**0.00**	**4.2**	3.01
CERES	**86.1**	**100.0**	**58.3**	1.4	0.61	**4.2**	**1.65**

B. Motion quality and computation

Controller	P ↑	SPL↑	Head↓	Path↓	S2 duty↓	Load↓
System 1 only	.348	.600	.504	2.521	0.0	.333
Always-on S2	.540	.650	.283	2.448	100.0	1.010
CERES	**.544**	**.802**	**.181**	**2.180**	64.4	**.611**

The capability-envelope behavior appears directly in the paired outcomes: CERES preserves all 48 routine System 1 successes while recovering 14/24 challenging trials. It therefore succeeds in 62/72 trials, improving over System 1 by 19.4 points (95% CI $[11.1, 29.2]$, Holm $p = .002$) and reducing collision by 15.3 points (CI $[-23.6, -6.9]$, $p = .013$). Always-on System 2 trades away eight routine successes while recovering 12 challenging trials; CERES retains those routine capabilities and completes two additional challenging cases, yielding a 13.9-point overall advantage (CI $[6.9, 22.2]$, $p = .023$) with 39.5% lower normalized expert-forward load. CERES also lowers collision density from 13.97 to 0.61 per 100 m, terminal heading error by 64.0%, and path ratio by 13.5%. The gain spans all three scene families, with 20/24, 22/24, and 20/24 successes in IndustrialFrame, OpenTank, and PipeCorridor, respectively (Appendix C).

These results expose the practical value of selective fast–slow coordination: CERES protects behavior already solved by the responsive policy, restores capability on difficult cases, and spends less deliberative computation than uniform escalation. The next study holds the shared admissibility and recovery contract fixed to isolate allocation choices. The complete CERES fast path reaches 45.23 ms at P99; Appendix G reports the separate expert-path timing distribution.

4.3 Capability-Aware Routing under Matched Closed-Loop Effectiveness

Table 2: Deployable router controls on the same 72 paired trials. Calls/dec. is routed expert-call intensity, and Forwards is the total expert-forward count.

Routing policy	Succ.↑	Chall.↑	Calls/dec.↓	Forwards↓
Frozen learned signal	80.6	41.7	1.514	39,385
Uncertainty threshold	83.3	50.0	2.204	51,545
Risk rules	**86.1**	**58.3**	1.992	33,739
CERES	**86.1**	**58.3**	**1.402**	**31,619**

With identical experts, safeguards, memory, task instances, admissibility, and recovery, CERES delivers lower deliberative allocation at matched closed-loop success. It matches risk rules at 62/72 overall and 14/24 challenging successes while reducing expert calls per decision by 29.6% and total expert forwards by 6.3%. Relative to uncertainty thresholding, it adds two successes with 36.4% fewer calls and 38.7% fewer forwards; relative to the frozen learned fast–slow signal, the complete CERES coordinator adds four successes with 7.4% fewer calls and 19.7% fewer forwards. CERES therefore realizes a favorable closed-loop effectiveness–computation frontier through selective slow computation. The router evidence centers on matched success with reduced deliberative computation; complete outcome statistics are reported in Appendix D.

4.4 Comparison with Adapted VLA Policies

Table 3: Comparison with adapted VLA policies on the same 72 paired tasks, action space, and outcome definitions. The three trained NTA configurations are shown individually; FLA reports the range across all seven frozen-backbone adaptations.

Policy	Regime	Succ.↑	Chall.↑	Coll.↓	P ↑	SPL↑
OpenVLA-7B	NTA	15.3	0.0	27.8	.200	.153
SmolVLA	NTA	15.3	0.0	72.2	.166	.113
Octo-Base-1.5	NTA	6.9	0.0	73.6	.142	.066
All seven FLA policies	Range	2.8–33.3	0.0–4.2	41.7–79.2	.012–.293	.000–.333
CERES	–	**86.1**	**58.3**	**1.4**	**.544**	**.802**

CERES exceeds the substantively trained NTA configurations by 70.8–79.2 success points and the FLA configurations by 52.8–83.3 under the common paired task, action, and outcome protocol. It

reaches 58.3% challenging-task completion, compared with at most 4.2% across the external policies. All paired success, collision, and progress tests remain significant after correction across the external policies; the NTA SPL tests are also significant (largest Holm $p = .0014$). These results establish the advantage of the complete CERES controller over the adapted generalist action policies. Appendix F reports every configuration separately together with complete failure, clearance, duration, training, and interface results.

4.5 Expert and Mechanism Ablations

Mechanism-targeted evaluation uses a pre-defined 112-trial population spanning 56 challenge classes with two seeds; each reported comparison contains 16 paired trials. Appendix E complements these targeted tests with complete leave-one-out evaluation over all 24 challenging deployment tasks.

Table 4: Mechanism-targeted paired comparisons; each contains 16 trials. The complete configuration grid is in Appendix E.

Target	Configuration	Succ.↑	Coll.↓	$P\uparrow$	SPL↑
State degradation	Fast only	0.0	31.3	.014	.000
	SR-directed	**87.5**	**0.0**	**.878**	**.859**
Action risk	Fast only	0.0	31.3	.022	.000
	SV	**43.8**	**0.0**	**.670**	**.438**
Plan revision	SV	31.3	0.0	.562	.311
	SV+CP	**37.5**	0.0	**.591**	**.373**
Temporal handoff	Memory disabled	0.0	0.0	.118	.000
	Full memory	**62.5**	0.0	**.847**	**.615**

The evidence supports a deliberate hierarchy of corrective roles. SR provides the strongest behavioral result, improving state-degradation success from 0/16 to 14/16 (87.5 points; CI $[68.8, 100]$, Holm $p = .002$). SV supplies current-action correction, reaching 7/16 successes and eliminating five collision episodes relative to the fast path on action-risk trials. CP extends correction across decisions: adding it to SV improves success from 5/16 to 6/16, and all 37 confirmed revisions are legal and first affect the next decision. Complete leave-one-out success and SPL estimates follow the same role ordering, while temporal memory raises success from 0/16 to 10/16 by converting realized outcomes into useful future evidence. Appendix E gives the familywise statistics and complete configuration grid. Configuration-dependent performance shows that corrective strength follows mechanism alignment, directly motivating CERES's selection of mechanism-appropriate write authority.

5 Conclusion

CERES reframes fast–slow autonomy from scheduling extra computation to governing corrective intervention. Capability-envelope coordination preserves responsive System 1 behavior where it is already competent and allocates System 2 where realized and current evidence indicate corrective value. Mechanism-specialized experts then bind that computation to representation repair, current-action verification, or subsequent-context revision with explicit causal scope. Across 72 paired trials, this design preserves every routine System 1 success, recovers 14/24 challenging tasks, improves overall success by 19.4 points, and reduces collision by 15.3 points; relative to always-on System 2, it achieves higher success with 39.5% lower normalized expert-forward load. Matched-success routing efficiency and mechanism-targeted evaluations further show that effective corrective computation depends on choosing the appropriate write authority and time of effect. This intervention-governance principle provides a practical path toward learned controllers that combine responsiveness, targeted reasoning, and closed-loop accountability.

AI USE STATEMENT

Generative AI tools were used solely for language polishing and reviewer-style critique to identify clarity and presentation issues. The research design, system implementation, experiments, analysis, and conclusions were produced and verified by the authors.

REPRODUCIBILITY STATEMENT

To support reproducibility, Section 4.1 and the appendix document the evaluation populations, paired protocol, controller interfaces, adaptation settings, metric definitions, statistical procedures, and causal invariants underlying the reported results.

REFERENCES

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14927–14937, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/aab085461de182608ee9f607f3f7d18f-Abstract.html.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert 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 *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert 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, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H. Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.025.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018. URL https://arxiv.org/abs/1801.08757.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning how hard to think: Input-adaptive allocation of LM computation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ff414825df833edb8b1839e3d5d495e9-Abstract-Conference.html.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.090.

Kay Hansel, Julen Urain, Jan Peters, and Georgia Chalvatzaki. Hierarchical policy blending as inference for reactive robot control. In *2023 IEEE International Conference on Robotics and Automation*, pp. 10181–10188, 2023. doi: 10.1109/ICRA48891.2023.10161374.

Wei Hung, Shao-Hua Sun, and Ping-Chun Hsieh. Efficient action-constrained reinforcement learning via acceptance-rejection method and augmented MDPs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/34b70ece5f8d273fd670a17e2248d034-Abstract-Conference.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. 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 *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Fanqi Lin, Ruiqian Nai, Yingdong Hu, Jiacheng You, Junming Zhao, and Yang Gao. OneTwoVLA: A unified vision-language-action model with adaptive reasoning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ebfabf372037aaa4a8d92c9b457ece3e-Abstract-Conference.html.

Mrinal Mathur, Barak A. Pearlmutter, and Sergey Plis. MIND over body: Adaptive thinking using dynamic computation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/955499a8e2860ed746717c1374224c43-Abstract-Conference.html.

Open X-Embodiment Collaboration. Open x-embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation*, pp. 6892–6903, 2024. doi: 10.1109/ICRA57147.2024.10611477.

Dung T. Phan, Radu Grosu, Nils Jansen, Nicola Paoletti, Scott A. Smolka, and Scott D. Stoller. Neural simplex architecture. In *NASA Formal Methods*, volume 12229 of *Lecture Notes in Computer Science*, pp. 97–114. Springer, 2020. doi: 10.1007/978-3-030-55754-6_6. URL https://doi.org/10.1007/978-3-030-55754-6_6.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011.

Swarnadeep Saha, Archiki Prasad, Justin Chih-Yao Chen, Peter Hase, Elias Stengel-Eskin, and Mohit Bansal. System 1.x: Learning to balance fast and slow planning with language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/605e59e78284907aa4fce5280838def3-Abstract-Conference.html.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. URL https://arxiv.org/abs/2506.01844.

Andy (DiJia) Su, Sainbayar Sukhbaatar, Michael Rabbat, Yuandong Tian, and Qinqing Zheng. Dualformer: Controllable fast and slow thinking by learning with randomized reasoning traces. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ed45d6a03de84cc650cae0655f699356-Abstract-Conference.html.

Pengzhi Yang, Haowen Liu, Monika Roznere, and Alberto Quattrini Li. Monocular camera and single-beam sonar-based underwater collision-free navigation with domain randomization. In *Robotics Research*, pp. 85–101. Springer Nature Switzerland, 2023. doi: 10.1007/978-3-031-25555-7_7.

Jianke Zhang, Xiaoyu Chen, Yanjiang Guo, Yucheng Hu, and Jianyu Chen. VLM4VLA: Revisiting vision-language-models in vision-language-action models. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1343e23bc2d34c054040e73ad86582cf-Abstract-Conference.html.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Tai Wang, Ya-Qin Zhang, Jingjing Liu, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/630d293833e09e1ecd892a898a20b074-Abstract-Conference.html.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Detailed System Specification

This appendix provides the implementation and evaluation detail needed to interpret and reproduce the results in the main paper. It follows the same distinction used throughout the paper: System 1 is the continuously available action policy, System 2 is conditional corrective computation, and SR, SV, and CP are three experts with different legal write targets. All reported quantities use frozen policies and pre-defined evaluation populations; checkpoint selection uses the designated development population, routing uses causal execution evidence, and every registered trial enters its aggregate.

A.1 Observation, State, and Action Interfaces

Table 5 specifies the information exposed to each part of the controller. RGB is a 240×320 color image. Sonar is represented by occupancy, normalized range, and validity channels. Point observations are transformed into the vehicle frame before valid-point pooling. The proprioceptive state and task command are embedded separately. Navigation context comprises the current locally available goal, an observed occupancy representation, a range scan, and explicit validity indicators. Missing observations use explicit availability masks. The policy interface is causal and observation-derived.

Table 5: Runtime interfaces and causal availability. "Current" denotes information available before selecting a_t; "next" denotes a value first available after executing a_t.

Quantity	Representation	Availability	Principal consumers
RGB	$240 \times 320 \times 3$ image	Current	State constructor, SR
Sonar	Occupancy, range, validity	Current	State constructor, SR
Point cloud	Valid vehicle-frame points	Current	State constructor, SR
Vehicle state	Proprioceptive vector	Current	State constructor, System 1
Task command	Text and goal condition	Current	State constructor, CP
Local context	Goal, occupancy, scan, masks	Current	System 1, admissibility checks
Risk evidence	Reliability and consistency terms	Current	Router, SR, SV
Diagnostic trace	EMA levels and first differences	Current	Router, CP
Outcome ρ_{t+1}	Progress, risk, realized mode	Next	Memory and later routing

The fused representation has dimension 128. RGB and sonar use convolutional encoders of width 32; point features use a shared encoder followed by masked maximum pooling; vehicle state uses an MLP; and task tokens are summarized recurrently. State and task queries attend to exteroceptive tokens, followed by a two-layer, four-head multimodal Transformer with zero dropout. Mean pooling produces h_t. The contrastive loss uses temperature 0.1 and aligns state, RGB, sonar, and point observations from the same physical instant; task information conditions fusion and reconstruction.

System 1 uses a causal history of at most eight decisions. Startup histories are left padded and masked. Its GRU hidden dimension is 256, with four-head causal attention and zero attention dropout. The action head applies componentwise affine scaling after a $\tanh$, ensuring $v_t \in [0, 0.85]$ m/s and $\omega_t \in [-1.2, 1.2]$ rad/s before the common acceleration-limited actuator interface. The evidential head produces independent normal-inverse-gamma parameters for the two action dimensions, and the induced predictive variance serves as routing evidence.

A.2 Capability Envelope and Legal Configurations

The router receives two complementary views of the current decision. A representation view summarizes h_t, task context, and perceptual reliability, while a diagnostic view summarizes action uncertainty, action inconsistency, failure evidence, the current capability, and realized-outcome memory. A learned elementwise gate combines the two encodings. The resulting fast–slow probability is evaluated jointly with local admissibility. Consequently, the learned score estimates corrective value, while deterministic checks preserve immediate action admissibility.

For completeness, Table 6 lists every expert configuration admitted by the coordinator. Every CP configuration includes SV, coupling plan revision with current-action certification. The empty configuration denotes the System 1 fast path, and the table lists the five corrective configurations.

Table 6: Admitted corrective configurations and their write semantics.

Configuration	Representation at t	Executed action at t	Context first used at $t+1$
SR	Refined by SR	Recomputed System 1 action	Unchanged
SV	Unchanged	Certified/projected by SV	Unchanged
SR+SV	Refined by SR	Certified/projected by SV	Unchanged
SV+CP	Unchanged	Certified/projected by SV	Revised after confirmation
SR+SV+CP	Refined by SR	Certified/projected by SV	Revised after confirmation

The capability graph contains APPROACH, AVOIDRISK, ALIGN, STABILIZE, RECOVER, and RETREAT. Graph masking retains legal successor capabilities before normalization. An accepted CP proposal persists across consecutive observations and satisfies minimum dwell and cooldown intervals of eight decisions. The proposed local subgoal is transformed using the current observed pose and stored in a persistent world frame. It becomes available on the next decision, preserving current-action certification and subsequent-context revision as separate causal events.

Current-observation admissibility. Algorithm 1 specifies the inexpensive test used for direct System 1 execution. It evaluates the actuator-executed preview against the current valid range scan. Let R be robot radius plus physical margin and Δt the control increment. Starting from the previewed (v_0, ω_0), the test includes the current step and then repeatedly slews desired velocity to zero using the fixed linear and angular deceleration limits. If $D = \sum_j v_j \Delta t$ and $\Theta = \sum_j \omega_j \Delta t$ over those steps, it requires range at least $R + D$ within a forward cone of half-width $\min\{\pi, |\Theta| + \arcsin(\min\{1, R/(R+D)\})\}$. This discrete swept-corridor check uses current observed ranges and assigns conservative zero clearance to an empty valid cone.

Algorithm 1 Causal fast-path admissibility test

Require: Current observation and valid scan, visible capability and lifecycle, System 1 action $a_t^{(1)}$, actuator-executed preview $\bar{a}_t$, and fixed actuator and footprint parameters.

Ensure: Fast-path admissibility $\chi_t(a_t^{(1)}) \in \{0, 1\}$.

$\mathcal{R} \leftarrow \emptyset$; identify the current causal local target q_t.
if capability or lifecycle is non-nominal, or progress has stalled **then**
    Add the contextual rejection reason to $\mathcal{R}$.
end if
if target, sensing, goal distance, or terminal alignment fails its fast-path check **then**
    Add the corresponding rejection reason to $\mathcal{R}$.
end if
if $a_t^{(1)}$ or $\bar{a}_t$ is nonfinite or outside the action bounds **then**
    return 0.
end if
Simulate the current preview and braking steps under fixed actuator slew limits; obtain travel D and absolute heading sweep $|\Theta|$.
Set $\alpha \leftarrow \min\{\pi, |\Theta| + \arcsin(\min\{1, R/(R+D)\})\}$.
Let d_{cone} be the minimum finite range of valid current scan rays in the forward cone of half-width α; assign 0 to an empty valid-ray set.
if $d_{\text{cone}} < R + D$ **then** add a corridor rejection reason to $\mathcal{R}$.
return $\mathbf{1}[\mathcal{R} = \emptyset]$.

A.3 EXPERT ARCHITECTURES AND COORDINATION

SR consumes the frozen foundation representation, raw encoded modalities, risk evidence with availability masks, and capability context. Modality embeddings have dimension 32 and the shared hidden dimension is 256. Its two heads predict a bounded residual representation and four modality-confidence values. The training target for the residual is the clean frozen representation of the same

underlying observation; the confidence targets follow the known observation degradation. When SR is the sole selected expert, its effective representation is restricted to a trust region around the foundation representation.

SV and CP use distinct role-specific 256-dimensional trunks. SV consumes the active action proposal, effective representation, seven-dimensional risk vector, action inconsistency, failure evidence, and capability context. It predicts current-action validity, a next-decision veto auxiliary, multi-horizon risk auxiliaries at 1, 5, 10, and 20 decisions, and as many as eight linear constraints. Unit-normalized constraint normals and signed offsets define the feasible action set used in Eq. 4. CP consumes task context, the effective representation, failure evidence, the six-dimensional risk trend, and current capability. Separate heads predict revision probability, graph-masked successor capability, and a two-dimensional local subgoal.

Selected experts are applied in the fixed order SR, SV, and CP. An SR update causes the frozen System 1 policy to recompute its proposal from the revised last latent state. A CP context confirmed on an earlier decision may condition an action-correction prior, and a scalar gate blends that prior with the current proposal. SV is applied after this fusion and is the final learned authority over the current action whenever its pathway is selected. The resulting command is clipped to the common action box and passed through the common actuator model.

A.4 Causal Decision and Routing Procedures

Algorithm 2 gives the decision-time procedure implemented by the frozen controller. It makes explicit two temporal boundaries: memory retrieval uses previously realized consequences, and a CP revision proposed at decision t first affects the subsequent decision. The legal configuration set additionally guarantees that every CP invocation is accompanied by SV.

Algorithm 2 CERES causal decision procedure

Require: Arrived history $\mathcal{H}_t$, private memory bank B, visible context c_t, capability graph G, and legal configurations $\mathcal{C}$.
Ensure: Executed action a_t, pending outcome record, and any next-decision CP revision.

Activate an earlier confirmed CP revision at its visibility time t.
Encode the arrived observation and history; obtain h_t, bounded $a_t^{(1)}$, and competence diagnostics.
for each executed action whose consequence arrived before t **do**
    Write its realized outcome to B.
end for
Retrieve capability-compatible events; form evidence e_t and infer routing scores.
Apply the causal admissibility contract to obtain $\zeta_t \in \{\emptyset\} \cup \mathcal{C}$.
Set $(\tilde{h}_t, \tilde{a}_t) \leftarrow (h_t, a_t^{(1)})$.
if $\mathrm{SR} \in \zeta_t$ **then**
    Set $\tilde{h}_t \leftarrow \mathrm{SR}(h_t, e_t)$; recompute $\tilde{a}_t$ with frozen System 1.
end if
Blend the visible plan-conditioned prior with $\tilde{a}_t$.
if $\mathrm{SV} \in \zeta_t$ **then**
    Certify/project $\tilde{a}_t$ onto the learned feasible set; set a_t to the result.
else set $a_t \leftarrow \tilde{a}_t$.
end if
if $\mathrm{CP} \in \zeta_t$ **then**
    Predict a successor capability and local subgoal.
    Register a pending revision after graph and lifecycle validity checks.
end if
Apply common bounds and actuator preview; use a stationary safe hold when the admissible set is empty.
Execute a_t or the hold; observe its consequence and update the pending outcome ledger.
Expose a confirmed CP revision beginning at decision $t + 1$.

The router supervision uses matched causal continuations for every legal configuration, including the empty fast-path configuration. Algorithm 3 is the exact selection logic underlying Eq. 3. A certified infeasible action receives its own abort outcome. Collision-free continuations dominate the candidate pool whenever one exists, so safety takes precedence over the compute term. The near-tie tolerance is $\epsilon_J = .05$, and an upgrade rescues an unsafe fast continuation or improves utility by at least $\delta = .10$.

Algorithm 3 Cost-aware target construction for capability-envelope routing

Require: Matched realized continuations $O_t(c)$ for $c \in \{\emptyset\} \cup \mathcal{C}$, utility J from Eq. 3, near-tie tolerance $\epsilon_J = .05$, and upgrade margin $\delta = .10$.
Ensure: Fast–slow, typed-configuration, and benefit targets, with regret for every configuration.

for each $c \in \{\emptyset\} \cup \mathcal{C}$ **do**
    Read capability completion, collision, certified abort, clipped progress, and $|c|$ from $O_t(c)$.
    Record a certified abort in its dedicated outcome category; evaluate $J(c)$.
end for
Let $\mathcal{F}$ contain all collision-free configurations.
if $\mathcal{F} = \emptyset$ **then** set $\mathcal{F} \leftarrow \{\emptyset\} \cup \mathcal{C}$.
Set $J^* \leftarrow \max_{c \in \mathcal{F}} J(c)$.
Set $\mathcal{T} \leftarrow \{c \in \mathcal{F} : J^* - J(c) < \epsilon_J\}$.
Choose c^* from $\mathcal{T}$ by minimum expert count, then fixed legal order.
Set $r(c) \leftarrow J^* - J(c)$ for every $c \in \{\emptyset\} \cup \mathcal{C}$.
Set $s \leftarrow \mathbf{1}$[fast continuation collides and c^* is collision-free].
Set $y_{\text{slow}} \leftarrow \mathbf{1}[c^* \neq \emptyset \ \wedge \ (s = 1 \ \vee \ J(c^*) - J(\emptyset) \geq \delta)]$.
return y_{slow}, c^*, the benefit target, and regrets $r(c)$ for auxiliary supervision.

A.5 Experimental Platform and Runtime

All CERES data generation, training, internal controls, and closed-loop evaluation use the same audited Linux server environment summarized in Table 7. The simulator and learned modules use explicit physical-GPU assignments. Paired policy variants use single-GPU rendering, spatial antialiasing, and restoration of the same physical, observation, random-state, and renderer-independent parent snapshot. External policy workers use model-specific isolated environments with the same physical server, simulator, task runtime, and GPU-lane constraints.

Table 7: Audited experimental platform for CERES training and closed-loop evaluation.

Component	Configuration
Server operating system	Linux 6.8.0-124-generic, x86-64, glibc 2.35
Processor and memory	AMD Ryzen 9 3900X (12 cores/24 threads); 62 GiB RAM
Accelerators	Two NVIDIA GeForce RTX 3090 GPUs, 24 GiB each; NVIDIA driver 535.309.01
Simulator	NVIDIA Isaac Sim 4.5.0-rc.36 (release build 19112.f59b3005)
Learning stack	CPython 3.10.20, PyTorch 2.5.1 with CUDA 12.1, NumPy 1.26.4
Simulation configuration	GPU physics; single-GPU rendering; spatial FXAA for paired evaluation
Control protocol	0.05 s physics increment, at most 960 decisions per trial; policy computation pauses simulation time
Timing protocol	One RTX 3090, explicit CUDA synchronization, observation-ready to final-action boundary

B Optimization and Data

B.1 Training Populations and Freezing Order

The core training population contains 700 episodes and 198,721 observation frames. A disjoint 100-episode development population supports offline evaluation and checkpoint selection where explicitly stated, while the 72 deployment trials form the held-out evaluation population. Deterministic episode-local block shuffling preserves short temporal neighborhoods while presenting every training frame exactly once per epoch. Table 8 reports the principal optimization settings. Component names identify the learned object.

All component-wise runs use gradient clipping at norm 1.0 and AdamW coefficients $(0.9, 0.999)$ with $\epsilon = 10^{-8}$. State-constructor masking is deterministic as a function of frame, epoch, and random seed. The state constructor is frozen before System 1 is fitted. Both are then frozen before training the corrective operators, so SR learns a representation intervention for a stable fast policy and SV learns constraints for a stable action interface. Router initialization uses chronologically

Table 8: Principal optimization settings. "Fixed final" selects the last pre-specified epoch.

Component	Epochs	Batch	Learning rate	Weight decay	Optimizer	Selection
State constructor	20	32/GPU	3×10^{-4}	10^{-4}	AdamW	Fixed final
System 1	30	32/GPU	3×10^{-4}	10^{-4}	AdamW	Fixed final
State Refiner	8	64	3×10^{-4}	10^{-4}	AdamW	Fixed final
SV and CP	8	64	3×10^{-4}	10^{-2}	AdamW	Fixed final
Router initialization	8	1	3×10^{-4}	10^{-2}	AdamW	Fixed final
Coordinated optimization	12	64	See text	—	AdamW	Fixed final

ordered continuation records and matched counterfactual outcomes, with two warm-start epochs followed by the pre-specified eight-epoch fit.

Coordinated optimization alternates two updates over 12 epochs. The first updates the router and learned memory readout at learning rate 10^{-4}. The second updates SV, CP, the correction prior, and coordination gate at learning rate 2×10^{-5}. A cosine schedule is applied per alternating cycle. The state constructor, System 1, and SR remain fixed. This asymmetric schedule adapts the coordinator and downstream corrective path jointly while preserving the representation and default action that define the intervention problem.

B.2 Objectives

The state-construction objective is

$$\mathcal{L}_{\text{state}} = \mathcal{L}_{\text{rec}} + 0.1\mathcal{L}_{\text{ctr}}, \tag{5}$$

where the reconstruction target is a stop-gradient summary of the unmasked modality. System 1 minimizes a smooth action loss together with the evidential negative log likelihood and evidence regularization,

$$\mathcal{L}_{\text{S1}} = \mathcal{L}_{\text{act}} + 0.1\mathcal{L}_{\text{NIG}} + 0.01\mathcal{L}_{\text{reg}}. \tag{6}$$

SR minimizes $\mathcal{L}_{\text{SR}} + 0.25\mathcal{L}_{\text{conf}}$, where the first term is squared error to the clean effective representation and the second is binary cross entropy on soft modality-confidence targets.

SV supervision combines current safety classification, learned constraint geometry, next-decision veto, multi-horizon risk, and projected-action approximation. CP uses separate valid masks and denominators for revision-event classification, legal successor classification, and revised-subgoal regression. Router supervision contains fast–slow, configuration, benefit, risk, memory-consistency, and specialization terms. The continuation utility in Eq. 3 prioritizes collision-free configurations and then applies the performance–cost objective among them. Safety therefore dominates the compute penalty.

B.3 Causal Memory

Memory is partitioned by capability and stores at most 128 entries per capability with first-in, first-out eviction. Retrieval uses squared Euclidean distance over conflict, anomaly, and failure evidence; the 16 nearest entries are aggregated with temperature 1.0. The memory embedding dimension is eight, its exponential return discount is .99, and output smoothing is .05. An entry is written after the corresponding action has executed and its consequence is observed. Each policy variant receives an identical initial bank, after which variant- and trial-private writes preserve comparison isolation.

C Closed-Loop Evaluation Protocol

C.1 Task Population and Pairing

The deployment study contains 72 trials: 48 routine and 24 challenging tasks, balanced across *IndustrialFrame*, *OpenTank*, and *PipeCorridor* with 24 trials per scene family. The challenging subset contains eight trials per scene family. Route families include direct approaches, left and right detours, outer arcs, narrow passages, and slalom-like routes. Tasks require both reaching a goal region and terminal heading alignment.

Every paired controller receives the same task command, scene, route, random seed, physical initial state, observation initial state, disturbance schedule, and 960-decision horizon. Initial observations are cached at the pairing boundary. Random-state and controller-history resets occur before every rollout. Every registered trial remains in the analysis, including collision, timeout, and low-progress outcomes. Physics advances by 0.05 s per integration step and pauses while the policy is evaluated; policy timing is therefore reported separately in wall-clock milliseconds.

A successful rollout requires the vehicle to remain within 0.78 m of the goal and 0.08 rad of the requested heading while stationary for eight consecutive decisions. Collision and horizon exhaustion are mutually recorded terminal failure modes. All commands share the same linear and angular bounds and the same acceleration-limited actuator. The observed map and range scan supply local feasibility evidence.

Table 9: Success counts by task stratum and scene family in the paired 72-trial deployment population.

Stratum	n	System 1 only	Always-on System 2	**CERES**
Routine	48	**48**	40	**48**
Challenging	24	0	12	**14**
IndustrialFrame, all	24	16	15	**20**
OpenTank, all	24	16	21	**22**
PipeCorridor, all	24	16	16	**20**
All trials	72	48	52	**62**

Table 10: Extended system-level outcomes omitted from the compact main-paper table. Unsafe-E and Unsafe-D are unsafe-episode and unsafe-decision rates; Head is terminal heading error and Path is traveled-distance ratio.

Controller	Coll./100m↓	Unsafe-E↓	Unsafe-D↓	Head↓	Path↓
System 1 only	13.97	18.1	1.76	.504	2.521
Always-on System 2	**0.00**	**4.2**	3.01	.283	2.448
CERES	0.61	**4.2**	**1.65**	**.181**	**2.180**

C.2 Metric Definitions

Let N be the number of episodes, T_i the number of executed decisions in episode i, $S_i \in \{0, 1\}$ its success indicator, C_i its collision-event count, L_i its traveled distance, D_i its shortest start-to-goal distance, and d_i^0, d_i^T its initial and final goal distances. Let U_i count selected System 2 decisions and F_i count completed SR, SV, and CP neural forwards, including diagnostic forwards. The reported metrics are defined in Table 11. Percentages are averages of binary indicators unless noted otherwise. Availability masks represent unobserved physical quantities as missing values.

SPL is defined as zero for an unsuccessful episode. Path-ratio and heading summaries use the protocol-valid episode set specified for each table. Normalized expert load includes both diagnostic and selected-expert forwards. System 2 duty and load therefore answer different questions: how often deliberation is selected and how much expert computation it performs.

Table 11: Definitions and interpretation of reported metrics.

Metric	Definition	Interpretation
Success	$N^{-1}\sum_i S_i$	Completion and terminal alignment
Collision	$N^{-1}\sum_i \mathbf{1}[C_i > 0]$	Episodes containing a collision
Collision density	$100\sum_i C_i / \sum_i L_i$	Events per 100 m traveled
Normalized progress	$N^{-1}\sum_i \mathrm{clip}_{[0,1]}((d_i^0 - d_i^T)/\max(d_i^0, 10^{-12}))$	Goal-directed displacement, including failures
SPL	$N^{-1}\sum_i S_i D_i / \max(D_i, L_i)$	Success weighted by path efficiency
Path ratio	$L_i / \max(D_i, 10^{-12})$	Traveled-to-shortest distance ratio
Heading error	Absolute terminal yaw error	Final alignment quality
Minimum clearance	Minimum observed obstacle distance	Worst local geometric margin
Unsafe episode rate	Fraction with any unsafe action event	Episode-level learned safety incidence
Unsafe decision rate	Unsafe decisions divided by decisions	Decision-level learned safety incidence
System 2 duty	$N^{-1}\sum_i U_i/T_i$	Episode-averaged intervention frequency
Expert load	$N^{-1}\sum_i F_i/(3T_i)$	Mean normalized forward load
Latency	Ready causal observation to final action, with synchronization	Ready-observation policy boundary

C.3 Statistical Analysis

All primary comparisons are paired at the trial level. Binary endpoints use two-sided exact McNemar tests. Continuous endpoints use two-sided paired permutation tests. We report 95% paired bootstrap intervals from 10,000 resamples. Holm correction is applied within each pre-specified family. The router family contains four controls crossed with success, collision, normalized progress, and SPL, for 16 tests. The external native-adaptation family contains three policies crossed with the same four endpoints. Expert removal and targeted-mechanism families are corrected separately.

The main system comparison yields a 19.44-point success improvement for CERES over System 1 (95% CI $[11.11, 29.17]$, Holm $p = .00195$), a 15.28-point collision reduction (CI $[-23.61, -6.94]$, Holm $p = .0127$), and a $.1956$ normalized-progress improvement (Holm $p = .00170$). Against always-on System 2, the success improvement is 13.89 points (CI $[6.94, 22.22]$, Holm $p = .0234$). The paired intervals and tests preserve common trial identity.

D Additional Routing Results

The router-isolation study replaces the pre-recovery routing proposal while keeping experts, memory initialization, admissibility checks, bounded recovery, task population, initial state, random state, and action semantics unchanged. The uncertainty control invokes the complete expert path at uncertainty above $.5$. The risk-rule control maps fixed thresholds on uncertainty, cross-modal conflict, anomaly, and failure evidence to legal configurations. The frozen learned control retains the supported binary M0/M2 fast–slow signal under the shared guard, whereas the complete CERES coordinator combines that signal with contextual legal-configuration selection. The frozen-distribution control shuffles within each trial the realized CERES configuration multiset, providing an outcome-informed diagnostic of allocation frequency and temporal placement.

Table 12: Router-isolation outcomes and computation on the same 72 paired trials. Frozen distribution is an outcome-informed timing diagnostic; Forwards is the unnormalized total of completed expert forwards.

A. Registered closed-loop outcomes

Router	Succ.	Chall.	Timeout	P	SPL
Frozen learned signal	58 (80.6)	41.7	12	.528	.752
Uncertainty threshold	60 (83.3)	50.0	11	.575	.778
Risk rules	**62 (86.1)**	**58.3**	10	.594	.823
Frozen-distribution diagnostic	64 (88.9)	66.7	8	.594	.842
CERES	**62 (86.1)**	**58.3**	9	.544	.802

B. Deliberative computation

Router	Decisions/ep.	Calls/dec.	Forwards
Frozen learned signal	254.82	1.514	39,385
Uncertainty threshold	270.31	2.204	51,545
Risk rules	249.24	1.992	33,739
Frozen-distribution diagnostic	232.68	1.702	30,001
CERES	—	**1.402**	31,619

CERES and risk rules both achieve 62/72 successes, while CERES reduces expert calls per decision from 1.992 to 1.402 and total completed expert forwards from 33,739 to 31,619. Relative to uncertainty thresholding, CERES adds two successes with $.802$ fewer calls per decision and 19,926 fewer forwards. Relative to the frozen learned binary signal, the complete coordinator gains four successes while reducing both call intensity and total forwards. Router-call intensity and total completed forwards use different accounting from the normalized expert load in Table 1; together they characterize selective computation under the shared execution contract.

The registered router-outcome family uses Holm correction. Under the shared corrective stack, CERES establishes a matched-success allocation advantage over the deployable risk rule, while the paired full-system and mechanism-targeted studies establish the behavioral and typed-intervention evidence.

E Expert-Specific Evaluation Detail

The mechanism-targeted population contains 56 pre-defined challenge classes, each evaluated with two seeds, for 112 trials. Every comparison row in Table 14 contains 16 paired trials. Challenge classes are grouped by the variable that must be corrected: degraded state evidence, imminent action risk, persistent plan failure, combined state and safety difficulty, combined safety and planning difficulty, all three mechanisms, and delayed evidence requiring temporal handoff.

The largest targeted effect is representation correction: SR-directed control succeeds in 14/16 state-degradation trials versus 0/16 for the fast path, an 87.5-point improvement (95% CI $[68.8, 100.0]$, Holm $p = .00195$). On action-risk trials, SV succeeds in 7/16 versus 0/16 and removes five collision episodes. On plan-revision trials, adding CP to SV raises success from 5/16 to 6/16 together with progress and SPL. CP also satisfies its intended intervention semantics: all 37 confirmed revisions select a legal graph transition, contain a valid local subgoal, and are first consumed on the next decision.

Complete expert removal evaluates all 24 challenging deployment tasks. Relative to CERES, removing SR reduces success from 14/24 to 9/24 and SPL from $.571$ to $.374$; removing the SV-mediated pathway increases collision episodes from one to four; and removing CP reduces success to 12/24 and SPL to $.487$. These complete-population effect estimates complement the targeted evidence.

Configuration-dependent results support typed specialization. SR is preferable on some representation-dominated cases, while SV+CP is preferable on some safety-and-planning cases.

Table 13: Complete leave-one-out results on all 24 challenging deployment tasks.

Configuration	Succ.↑	Coll.↓	TO↓	P ↑	SPL↑
SR removed	37.5	0.0	62.5	.550	.374
SV-mediated pathway removed	45.8	16.7	**37.5**	.589	.428
CP removed	50.0	0.0	50.0	.592	.487
CERES	**58.3**	4.2	**37.5**	**.610**	**.571**

Configuration	Min. clear. (m)↑	Mean clear. (m)↑	Time (s)↓
SR removed	**.222**	.685	36.09
SV-mediated pathway removed	.161	**.808**	**26.41**
CP removed	.212	.595	34.18
CERES	.187	.678	29.76

Table 14: Complete mechanism-targeted configurations; each row contains 16 paired trials.

Target	Configuration	Succ.↑	Coll.↓	P ↑	SPL↑	Clear.↑
State degradation	Fast only	0.0	31.3	.014	.000	.959
	SR-directed	**87.5**	**0.0**	**.878**	**.859**	.324
Action risk	Fast only	0.0	31.3	.022	.000	.656
	SV	**43.8**	**0.0**	**.670**	**.438**	.200
Plan revision	SV	31.3	0.0	.562	.311	.229
	SV+CP	**37.5**	0.0	**.591**	**.373**	.229
State+safety	Fast only	0.0	56.3	.033	.000	.313
	SR	**87.5**	0.0	**.932**	**.852**	.199
	SV	56.3	0.0	.721	.559	.222
	SR+SV	50.0	0.0	.660	.496	.236
Safety+planning	SV	31.3	0.0	**.652**	.313	.248
	SV+CP	**37.5**	0.0	.627	**.375**	.260
All mechanisms	Fast only	0.0	43.8	.020	.000	.455
	SR+SV	37.5	0.0	.610	.375	.209
	SV+CP	**43.8**	0.0	**.672**	**.399**	.194
	SR+SV+CP	31.3	0.0	.556	.313	.203
Temporal handoff	Memory disabled	0.0	0.0	.118	.000	.921
	Full memory	**62.5**	0.0	**.847**	**.615**	.288

The router therefore selects the intervention set whose write authority matches the active failure mechanism.

F External Policy Adaptation Details

The comparison contains seven distinct VLA backbones and ten policy–adaptation configurations because OpenVLA, SmolVLA, and Octo are evaluated under both NTA and FLA. NTA updates architecture-supported action-generating modules under each model's native objective; FLA holds every backbone fixed and fits the common action bridge. The comparison evaluates complete policies under common task dynamics, action semantics, and outcomes: CERES uses its full onboard multimodal interface, while each external policy uses its supported native inputs. This design preserves valid pretrained interfaces for RGB-based models. All external actions are converted to the same two-dimensional body-frame command and clipped to the same action box.

Population and budget accounting. All adaptations derive from the same 700 training episodes used to construct CERES's training archive, using outcome-independent episode retention. NTA selects 16 deterministic integer-spaced anchors per episode, including the first and last decisions,

for 11,200 samples. Effective batch 16 and 700 optimizer steps consume one complete pass through this episode-balanced population. FLA selects four anchors per episode (2,800 total) and solves its ridge objective in closed form. Checkpoint selection uses four equally constructed anchors from each of 100 disjoint development episodes; Paper72 remains the held-out evaluation population. The 198,721 CERES frames and these anchor counts are different optimization units: CERES uses modular frame/window objectives, whereas OpenVLA, SmolVLA, and Octo use native one-, 50-, and four-step action targets. Source-episode coverage, outcome-independent sampling, model selection, and paired deployment are therefore controlled explicitly.

Table 15: Complete per-policy outcomes underlying the compact FLA range in Table 3. NTA and FLA are alternative adaptation regimes.

Policy	Regime	Succ.↑	Chall.↑	Coll.↓	P ↑	SPL↑
OpenVLA-7B	NTA	15.3	0.0	27.8	.200	.153
SmolVLA	NTA	15.3	0.0	72.2	.166	.113
Octo-Base-1.5	NTA	6.9	0.0	73.6	.142	.066
OpenVLA-7B	FLA	33.3	0.0	47.2	.284	.333
SmolVLA	FLA	6.9	0.0	79.2	.150	.068
Octo-Base-1.5	FLA	31.9	4.2	45.8	.287	.319
SpatialVLA-4B	FLA	33.3	0.0	45.8	.293	.333
X-VLA	FLA	5.6	0.0	48.6	.156	.051
π_0	FLA	11.1	0.0	62.5	.166	.111
$\pi_{0.5}$	FLA	2.8	0.0	41.7	.012	.000
CERES	–	**86.1**	**58.3**	**1.4**	**.544**	**.802**

Table 16: Failure mode, clearance, and episode efficiency for the VLA comparison.

Policy	Regime	Coll.↓	TO↓	Clear. (m)↑	Time (s)↓
OpenVLA-7B	NTA	27.8	56.9	.794	29.88
SmolVLA	NTA	72.2	**12.5**	.353	16.46
Octo-Base-1.5	NTA	73.6	19.4	.305	15.01
OpenVLA-7B	FLA	47.2	19.4	.725	14.20
SmolVLA	FLA	79.2	13.9	.174	20.27
Octo-Base-1.5	FLA	45.8	22.2	.677	15.89
SpatialVLA-4B	FLA	45.8	20.8	.723	15.00
X-VLA	FLA	48.6	45.8	.493	26.84
π_0	FLA	62.5	26.4	.484	17.58
$\pi_{0.5}$	FLA	41.7	55.6	.853	29.67
CERES	–	**1.4**	**12.5**	**1.017**	**12.08**

F.1 Native Task Adaptation

All three NTA policies use validation every 100 steps and one fixed training seed and recipe. Their respective native development objectives select the checkpoint, and parameter-delta receipts verify that the designated modules changed.

Table 17: Native task-adaptation configurations.

Policy	Trainable portion	Parameters	LR	Batch	Steps
OpenVLA-7B	Rank-32 QLoRA over vision projector and language–action policy	110.8M	5×10^{-4}	16	700
SmolVLA	Action expert plus state, action, and time projections	99.9M	10^{-4}	16	700
Octo-Base-1.5	New 2-D action head and supported non-Transformer modules	7.1M	3×10^{-5}	16	700

OpenVLA uses NF4 quantization with bfloat16 computation, LoRA scale 16, zero LoRA dropout, AdamW, and zero weight decay. SmolVLA keeps its vision and vision-language backbone frozen, trains its native action expert and projections with AdamW, 70 warmup steps, and effectively zero weight decay. Octo keeps its Transformer frozen and trains the new action space with AdamW, 70 warmup steps, and weight decay .01. Measured nonzero parameter changes verify that all three adaptations update the intended modules. Their improved development losses and selected checkpoints verify substantive task adaptation.

F.2 Frozen-Backbone Linear Action Adaptation

The second regime fits a two-dimensional ridge action bridge on 2,800 temporally valid samples from the same 700 training episodes, using regularization 10^{-3}. The pretrained representation backbones of OpenVLA, SmolVLA, Octo, SpatialVLA, X-VLA, π_0, and $\pi_{0.5}$ remain fixed. The ridge bridge directly calibrates the final command interface, whereas native adaptation optimizes model-specific action objectives; the two regimes therefore test complementary adaptation strategies.

All external policies receive current RGB, task text, and body-frame goal information; SmolVLA additionally receives its supported current proprioceptive input. These current causal interfaces define the complete action-generation strategies evaluated in Table 3.

G Timing Method

Latency is measured from an already available causal observation to the final action returned by the policy, with CUDA synchronization at both timing boundaries. This ready-observation boundary measures policy computation. All fixed timing cases use one RTX 3090 and the same software environment.

Table 18: Detailed ready-observation policy latency in milliseconds.

Path	n	Mean	Median	P95	P99	> 50 ms
Standalone System 1	74	32.61	32.70	33.45	33.75	0.0%
CERES fast path	411	42.30	42.22	43.59	45.23	0.0%
CERES expert decision	1507	203.81	170.83	358.21	470.10	100.0%

Standalone System 1 measures the default policy, whereas the CERES fast path includes the evidence and coordination used to select direct execution. CERES remains below 50 ms at P99 for this complete fast path. Synchronous expert decisions provide the additional corrective computation reported in the third row, and the 0.05 s value denotes the simulator integration interval.

H Causal and Reproducibility Invariants

Table 19 summarizes the invariants applied to every internal comparison. They separate learned intervention selection from fixed execution legality and preserve consistent information access under component removal.

The bounded recovery behavior follows the same execution-consistent interface. When the feasible-command set is empty, the vehicle holds translation and uses the latest observed scan and occupancy context to search for a locally admissible alignment or escape direction. Any proposed motion passes through SV when its pathway is active. This shared behavior provides a common liveness and safety contract across compared routing policies using current onboard evidence.

Together, these details make explicit how the two central design ideas are instantiated and tested. System 1/System 2 coordination is evaluated by pairing selective computation with fast-policy and always-on controls, then isolating alternative routing proposals under a common corrective stack. Three-expert specialization is evaluated through legal write targets, complete expert removals, mechanism-targeted trials, temporal-memory controls, and direct verification of next-decision planning semantics. Matched action bounds, information access, trial selection, and evaluation procedures ground the closed-loop intervention claims.

Table 19: Causal and comparison invariants.

Invariant	Enforcement
Causal information boundary	Routing and expert inputs are available at or before decision t; realized outcomes enter subsequent decisions.
Delayed planning effect	A CP revision confirmed at t is first visible at $t + 1$.
Action authority	Whenever selected, SV is the final learned source of the current executed command.
Legal capability changes	Graph masking, persistence, dwell, and cooldown checks precede a CP commit.
Common actuation	Every configuration uses identical action bounds and acceleration limits.
Paired initial state	Task, seed, physical state, observation state, disturbance schedule, and horizon are shared within a pair.
Variant-isolated memory	Initial memory is copied; subsequent writes remain private to each variant and trial.
Runtime-input integrity	Runtime inputs comprise causal observations, task context, and realized controller history.
Observation-derived geometry	Local admissibility uses current observed occupancy and range evidence.
Faithful expert removal	Removing an expert removes its corresponding write operation.
Complete outcome retention	Collisions, timeouts, and low-progress trials remain in every aggregate.
Explicit missingness	Availability masks preserve unobserved quantities as missing values.