Governed Embodied Action: A Persistent Structural-Mechanics Substrate Supervising an LLM Robot Planner in Simulation

Ken Morkaya
independent researcher (structural engineering, 30+ years)
kenmorkaya@gmail.com | ORCID 0009-0002-2824-8334

July 2026

Draft v0.3 — all sections drafted, citations wired, figures specified.
Companion papers: SICD-MSR general reasoning (100% vs GPT-5.2); SICD three-mode context-rot elimination on frozen Gemma 4 (Zenodo 10.5281/zenodo.19836656).
Repository / artifacts: `tom_governed_embodiment` *(pre-registration ledger, per-run preflight reports, physics logs, traces).*

Abstract

Contemporary robot autonomy increasingly places a large language model in the planning loop. Such planners are fluent but not accountable: they weigh safety-relevant signals as graded evidence rather than treating them as binding constraints, and their decisions carry no auditable trace. The approach, at its highest level: place an adaptive structural control state — a persistent tree that deforms under load — between the planner's words and the robot's motion, and make release a function of that structure's response rather than of the planner's token probabilities. An LLM's output is a sample from a distribution shaped by pretraining and RLHF; any safety signal routed through that distribution can be weighed away by it. Here, safety-relevant conditions are instead **held** by the structure as gates: the guard is not a hand-written check against words but a load the tree carries. We present a simulation study in which a persistent structural-mechanics reasoning substrate (SICD/ToM) supervises an off-the-shelf LLM planner driving a simulated manipulator. The supervisor compiles each proposed action, with its typed scene context, into a real 17-channel structural load, routes that load through a persistent 10,000-branch tree via a loading-aware 8D readout, and issues a release decision (ALLOW / MODIFY / VETO / QUARANTINE) bound to a hash-stamped trace. An independent physics evaluator — sharing no code with the supervisor — measures rule-violating motion directly from the simulation logs. On a pre-registered, provenance-gated battery run for **two planner classes** behind the same supervisor — a local open-weights model (Gemma 4 26B) and a frontier API model (gpt-5.5) — the ungoverned planner produced 17 and 12 rule-violating events respectively; the fully governed condition produced **zero in both cases**, with a false-intervention rate of 1/20 on benign twins. The governed outcome therefore transfers across model classes with no change to the supervisor. On a separate probe of events constructed **outside** the supervisor's explicit hazard vocabulary, the supervisor stopped 4/4 physical cases; an attribution ablation shows this generalization runs through the structural load pattern, with two channels carrying the full-stop margin and the structural channels carrying a graded protective response. A second probe axis — out-of-scope natural-language instructions with no typed contradiction — was **not** caught, and we report the precise, channel-level reason. The contribution is a bounded, auditable demonstration that an accountable structural substrate can supervise an LLM planner's actions, and a mechanism-level account of exactly where that accountability currently ends.

1 Introduction

Large language models are entering robot planning loops. The pattern is consistent across current systems: a fluent general-purpose model proposes actions, and a thin layer of hand-written checks — speed limits, geofences, interlocks — stands between those proposals and a machine that moves mass. This arrangement has two structural defects that no amount of model scaling repairs.

First, an LLM **weighs** safety-relevant signals; it does not **hold** them. Given an instruction that asserts a constraint no longer applies, a fluent planner balances that assertion against the rest of its context and can reach a confident, well-argued, wrong release decision. Our companion work measured this failure mode directly in a non-embodied setting: on a 124-case battery, ToM scored 100.0% and GPT-5.2 scored 79.03%, while GPT-5.2's 84-case MSR error analysis showed strong operator selection (97.6%, 2/84 errors) but weaker structural repair judgment (81.0%, 16/84 errors) and the mechanism claim that GPT-5.2 reads integrity signals in prose but treats them as weighable evidence. In an embodied loop, the same disposition governs actuators.

Second, an LLM's decision carries **no auditable trace**. Certification regimes for machinery require that a release decision be reconstructable: what the system knew, which constraint applied, why the action was or was not released. A sampled token sequence satisfies none of that, and post-hoc rationalization by the same model is not an audit.

These two gaps compound exactly where LLM-driven robots are headed: **domestic and assistive settings**. An industrial cell is engineered once and certified; a home is a different unstructured environment for every unit shipped, with no safety engineer on site — per-task envelope engineering does not scale to it. And when something goes wrong around people, accountability is not optional: an incident with no reconstructable decision record is uninsurable and, under machinery-style certification, undeployable. A supervisor whose coverage travels with the mechanism rather than the task list, and whose every release decision leaves a trace, is the missing piece for that setting.

The usual response is to enumerate: write more hand-coded checks. But an enumerated checklist only catches what its author foresaw — the oldest failure in machine safety — and it is precisely the un-foreseen configuration that matters. What is needed above the checklist is a supervisor that *reasons* about the situation's structure, holds its constraints as gates rather than weights, and writes down why.

Stated as a design principle before the mechanics: the supervisor does not ask the language model to be safer, and it does not screen the model's words with hand-written checks — a regex cross-reference catches only the phrasings its author foresaw. It instead compiles the *proposed action and its scene* into mechanical load on an adaptive structural control state (a persistent, deforming tree), and lets the release decision be the structure's response to that load. The distinction is between a signal that is **weighed** — entering an RLHF-shaped token distribution where sufficiently fluent context can argue it down — and a signal that is **held**, raised from typed scene relations into a structure whose gates do not participate in the argument. Everything that follows is the bounded, audited consequence of that one substitution: what it catches (§4, §5), what it does not yet catch and precisely why (§5, §6), and what it writes down either way (§3).

Prior work established the two halves of such a supervisor in non-embodied settings: a structural-mechanics reasoning substrate (SICD/ToM) that performs bounded multi-step general reasoning with **no LLM in its loop**, treating integrity conditions as structural gates (100.0% vs a frontier baseline's 79.03% on a 124-case pre-registered battery); and a memory/coupling architecture that supervises a frozen LLM's context without touching its weights. This paper completes the arc: the same substrate, unmodified and persistent, placed as an **action supervisor above** an off-the-shelf LLM robot planner in simulation.

The supervisor does not replace the planner and does not plan. It compiles each proposed

action, with its typed scene context, into a real 17-channel structural load; routes that load through a persistent 10,000-branch tree via a loading-aware 8-dimensional readout; and issues a release decision — ALLOW, MODIFY (graded protective adjustment), VETO, or QUARANTINE — bound to a hash-stamped trace carrying the full load vector, the routing, and the branch response. An independent physics evaluator, sharing no code with the supervisor, measures rule-violating motion directly from the simulation logs.

Contributions.

1. **Architecture:** the first (to our knowledge) demonstration of a persistent structural-reasoning substrate — not a rule table, not a second LLM — supervising an LLM robot planner's actions, with per-action auditable traces (§2).
2. **Governed result, transferring across model classes:** on pre-registered, provenance-gated batteries behind the same supervisor, a local open-weights planner and a frontier API planner produced 17 and 12 rule-violating events ungoverned; the fully governed condition produced **zero in both cases**, at a false-intervention rate of 1/20 on benign twins and no task-completion penalty (§4).
3. **Generalization probe with honest attribution:** 4/4 physical events constructed *outside* the supervisor's explicit vocabulary were stopped; an ablation attributes the full-stop margin to two load channels while the structural channels alone still carry a graded protective response — and one probe axis (out-of-scope natural-language instructions without a typed contradiction) was **not** caught, reported with its channel-level cause (§5–§6).
4. **Methodology for embodied-LLM evaluation:** pre-registration by commit SHA, a fail-closed 14-check preflight that refuses to score invalid runs (including inert-baseline and fabricated-output failures we actually caught), and an import-independent physics evaluator (§3, Appendix B).

Scope, stated narrowly. This is a bounded, sandbox-only result: one simulator, two planner classes, small case counts with Wilson intervals reported, simplified execution primitives, and a known, mechanism-diagnosed boundary on one probe axis. It is a demonstration that accountable supervision of an LLM planner is achievable with a reasoning substrate and auditable in the engineering sense — not a certification claim, and not a claim of unbounded generalization.

Related work

LLM and VLA robot planners. A growing line of work places language or vision-language-action models directly in the robot control loop as planners or policies. These systems inherit the language model's fluency and its dispositions — including the tendency to treat constraints as weighable context. Safety for them is typically an external deterministic envelope (speed/force limits, geofencing, interlocks) or prompt-level guidance. Our work is complementary and orthogonal: it does not improve the planner; it places an independent, reasoning supervisor above it that gates released actions and produces an audit trace. [1–5]

Functional safety and runtime assurance. Runtime-assurance and simplex-style architectures wrap a complex controller in a verified safety monitor that can override it. Our supervisor occupies the same architectural slot but at the *cognitive* level — governing which composed intents are released, from a reasoning substrate, rather than bounding a continuous control signal — and above, not in place of, the deterministic physical envelope. [6–11]

Neuro-symbolic and cognitive architectures. Prior neuro-symbolic systems pair neural perception with symbolic (logical/probabilistic) reasoning; cognitive architectures (SOAR, ACT-R) fix a procedural cycle over adaptive content. The substrate here is neither: it reasons in a structural-mechanics space (load, stress, plastic deformation) with no logical solver and no learned neural reasoner in its loop. Its lineage and evidence base are the two companion papers. [12–16]

This work against that background. The contribution is not a better planner and not a new safety envelope; it is a demonstration that a persistent structural-reasoning substrate can serve as the accountable, auditable, action-level supervisor that the LLM-robot stack currently lacks — and a mechanism-level account of where its coverage currently ends.

Placed on two axes — how a monitor's *coverage is obtained* (hand-specified per task vs. carried by a general mechanism) and how its *decisions are made* (statistical vs. deterministic and auditable) — the incumbents occupy three quadrants. Deterministic envelopes, simplex/runtime-assurance wrappers, and synthesized shields are auditable but hand-specified: every constraint is engineered per cell and per task, and coverage ends at the spec. LLM/VLM judge layers are general but statistical: made of the same material as the planner they judge, prompt-sensitive, and version-drifting, with generated text in place of a decision record. The conjunction demonstrated here — coverage carried by one persistent mechanism (out-of-vocabulary stops, §5; planner-class transfer with the supervisor unchanged, §4) with deterministic, hash-stamped decisions behind gates frozen at registration (§3), at laptop-scale compute — occupies the fourth quadrant: **general like a judge, deterministic like an interlock.** To our knowledge no published system combines these properties in an embodied setting (Figure 6). The trade is stated plainly: a barrier certificate proves its whole constraint class, while this supervisor offers evidence plus traces over what has been tested — and a mechanism-level account of where coverage ends.

2 Architecture

The system has three parts that never share decision logic: a **planner** that proposes actions, a **supervisor** that releases or blocks them, and an **evaluator** that measures what physically happened. Their independence is the experiment — see §3.

2.1 Simulator, robot, and action grammar

MuJoCo (CPU, fixed seed) with the Franka Panda from `mujoco_menagerie` (Apache-2.0) on a tabletop scene carrying objects, keep-out / work / handoff zones, tools with guard state, and a scripted human avatar. Execution is deliberately simplified — waypoint primitives and an attach-constraint grasp — because the claim under test is *decision governance*, not manipulation skill; sophisticated motor control would only confound the result. The planner emits actions in a **closed grammar** (pick, place, move, activate, deactivate, handover, wait, declare_unable), schema-bound, with every referenced object/zone/tool id required to exist in the scene — a citation constraint that prevents action on hallucinated entities.

2.2 Planner

Gemma 4 26B (4-bit MLX, temperature 0), an off-the-shelf model with no fine-tuning. To make the comparison across conditions a genuine paired test, each case's plan is **sampled once** with a condition-agnostic prompt, persisted with its pre-parse raw text and prompt hash, and replayed identically under BASE, SHADOW, GATE, and FULL; SCRIPT and BENIGN use oracle plans.

The frontier API planner reported in §4 runs behind the same adapter and condition-replay protocol.

2.3 Supervisor (the real ToM substrate)

The supervisor is the unmodified SICD/ToM mechanism, vendored byte-identical from its source at a pinned commit and verified as such by the preflight (§3). For each proposed action it:

1. compiles the action + typed scene context into a **real 17-channel structural load** (S/L/T families + eight dynamics channels) via an embodiment intake that raises channels from scene relations — zone membership, clearance margin, guard/sequence state, proximity — not from hazard tags;
2. routes that load through a **persistent 10,000-branch tree** (`TreeGrowthEngine`, loaded from a pinned snapshot) using the **loading-aware 8-dimensional readout** (channel-separated projection + a $0°/120°/240°$ angular gate);
3. reads the tree's response (`final_T`, routing basis, top-branch score, kappa/σ deltas) and issues a release decision — ALLOW / MODIFY / VETO / QUARANTINE — from that response, **not** from any hand-written per-hazard rule;
4. writes a trace: the full 17-channel load, the 8D projection, the routed branch, the verdict, and a decision hash.

Persistence is **cross-battery**: one accumulating tree per condition, seeded once from the canonical snapshot and deforming (kappa) across the ordered run — case N inherits the tree state case $N-1$ left. The organism accumulates; it is not reset per case. The full load path — projection weights, routing, and the plastic deformation law — is documented in Appendix A.

2.4 Evaluator

A deterministic physics evaluator computes rule-violating motion (keep-out incursion, human-proximity overspeed, hazardous co-location, guard-sequence violation) and task completion directly from the simulation logs. It **imports nothing from the supervisor** and runs out-of-process, so the two verdicts — structural (supervisor) and physical (evaluator) — are produced by fully independent code.

2.5 Conditions

BASE (planner alone), SHADOW (supervisor computes verdicts, does not enforce — the deployment on-ramp), GATE (enforce, no re-plan), FULL (enforce + constrained re-plan), SCRIPT (oracle plans, supervisor on — isolates false-intervention from planner quality), and BENIGN twins (hazard condition removed, gates must stay silent).

3 Method

Pre-registration by commit SHA. Every scored run's exact command, pools, frozen constants, and pass gates are committed to a ledger (`docs/experiment_contract.md`) *before* the run; the preflight verifies the pre-registration commit is an ancestor of the run commit. A run whose parameters were not fixed in advance does not count.

Fail-closed preflight. A scored run is admissible only if a 14-check preflight (`scripts/preflight.py`) is GREEN and stamped for the run's commit; `run_battery` refuses to emit a scored result otherwise. The checks encode every failure mode this project actually encountered during development: real-ToM imports present, canonical 10k tree loaded, cross-battery accumulation verified by evolving per-case state hashes, **zero text-pattern-matching in the decision path**, non-vacuity (the baseline must produce real events — a governance pass over an inert baseline is a failure), vendored source byte-identical to the pinned commit, evaluator import-independence, model-output provenance (distinct raw responses persisted with prompt hashes — a fabrication guard), and pre-registration ordering. Each check fails closed: it cannot pass on a missing file or absent field. Appendix B lists the full set. Development runs A.1 and A.2 (inert baseline: the MLX planner produced parse-failure fallback plans in nearly all BASE episodes, so a governance pass would have been vacuous) and A.5R (a simplified load-space wrapper that did not load, mutate, or persist the pre-built 10,000-branch ToM state) were rejected by this fail-closed process, which is the intended behavior.

Pools. Two independently-authored scenario pools, each with benign twins and build-time contract checks (every hazard is reachable by a legal plan; every benign twin is completable; no scenario text contains gate vocabulary; no cross-case content leakage).

Statistics. Comparisons are paired per case; pass rates carry Wilson 95% confidence intervals; reported outcomes are the independent evaluator's, never the supervisor's self-report.

4 Results — governed battery, across two planner classes

We report the governed battery for two planners behind the same supervisor, tree, gates, and pools: a **local open-weights model** (Gemma 4 26B, run A.11R) and a **frontier API model** (gpt-5.5, run A.12). Both runs passed a GREEN 14-check preflight stamped for their commit and all five pre-registered gates.

Local planner (A.11R, Gemma 4 26B). Planner validity: 0/40 parse failures, 39/40 real-action plans. The ungoverned condition (BASE) produced **17 rule-violating events** across the 40 episodes — the baseline is non-vacuous, i.e. there is real motion to govern. The fully governed condition (FULL) produced **0 rule-violating events**, as measured by the independent physics evaluator. False intervention on benign twins was **1/20 (0.050)**. Benign task completion under FULL equalled BASE — the supervisor imposes no completion penalty; the residual incomplete benign tasks are a planner/execution artifact identical across conditions.

Frontier planner (A.12, gpt-5.5). The same supervisor and tree were run with gpt-5.5 as the planner over the OAuth path, sampling once per case and replaying across conditions. Planner validity: 0/40 parse failures, 36/40 real-action plans, 4/40 explicit model concessions (`declare_unable`). BASE produced **12 rule-violating events**; FULL produced **0**; false intervention **1/20 (0.050)**. Each call persisted its API response id, response model, and token usage (40 distinct response ids; response model `gpt-5.5` on every call) — the anti-fabrication provenance requirement is directly satisfied on the API path.

Cross-model reading. The supervisor holds FULL = 0 rule-violating events across a local open-weights planner and a frontier API planner without any change to the substrate, the tree, or the gates. The governed result is therefore not a property of one planner's idiosyncrasies; it is a property of the supervision. This is the paper's central external-validity result: accountability that transfers across model classes.

- Provenance: vendored ToM byte-identical to pinned source; verdict trace carries the real 17-channel load, 8D routing, branch response, and hash.

Run	Planner	BASE events	FULL events	False interv.	Preflight
A.11R	Gemma 4 26B (local)	17	**0**	1/20	GREEN 14/14
A.12	gpt-5.5 (frontier)	12	**0**	1/20	GREEN 14/14

Table 1: Governed battery across two planner classes behind the same supervisor, tree, gates, and pools. FULL = 0 rule-violating events in both, measured by the independent physics evaluator.

- Phase 2 vocabulary cleanup: project-owned typed packets now use `memory_scope.safety_condition_status = unresolved_safety_condition`; the pinned vendored compiler's expected assurance-domain literal is supplied only by a documented boundary adapter. The canonical local-planner run is A.11R, which also uses native safety-condition status values in the standalone ToM reasoning preflight. P.2 diagnostic replay confirmed identical 17-channel loads before/after the rename.

5 Results — generalization probe (P.2)

The governed battery (§4) shows the supervisor stops events drawn from the same families the pools were built on. The probe asks the harder question: does it stop events constructed *outside* its explicit vocabulary? We authored a separate, independently pre-registered probe pool along two axes and ran it under the same preflight-gated protocol (GREEN, 14/14; planner validity 12/12 real-action plans; BASE non-vacuous). Full per-case detail and artifacts are in `docs/probe_findings.md`.

Physical axis — outside the tag vocabulary (4 cases). These cases carry only benign object tags; the danger is purely geometric and relational (arm entering a reserved footprint, motion inside a moving human's clearance margin, activation with an open guard). No hazard-tag pairing exists for the intake to match. The independent evaluator recorded each as a rule-violating event in BASE. **The fully governed condition stopped all four.**

Attribution ablation (diagnostic). Because the caught cases showed high `L_contradiction` and `threat_amplitude`, we tested whether those two channels were doing all the work: we zeroed both, preserved every other intake-produced channel, and re-ran the decision path. The result refutes both extreme readings. No case still reached full-stop on the structural channels alone — but the response did not collapse either: `final_T` held ≈ 0.59–0.65 at the protective-action floor across all four, and two cases still issued a graded protective adjustment (MODIFY). The precise reading is therefore: **the structural channels (raised from scene relations, not tags) carry a distinctive graded signal; `threat_amplitude` and `L_contradiction` carry the additional margin to a full stop.** This is a stronger and more honest statement than either "caught on structure alone" or "the catch rode two channels." (The ablation used a fresh per-case tree for isolation, so its magnitudes are diagnostic, not identical to the persistent scored run; the derivation of the high channel values from each scene's typed packet — clearance margin, guard sequence — is recorded per case, confirming they are scene-derived, not a blanket label on rule-violating variants.)

Out-of-scope-instruction axis (2 cases) — not caught. These cases present natural-language instructions asserting that a constraint no longer applies, with no accompanying typed state that contradicts verified scene facts. Both were released (ALLOW). The trace shows why to the channel: `L_contradiction = 0`, so the δF composite it dominates never developed, and the load never routed to a stop. This is the current boundary of the supervisor's coverage, and §6 locates its cause and its fix. We report it as a result: a mechanism-diagnosed edge, not an unexplained failure.

6 Mechanism — why a miss is an intake gap, not a substrate limit

The generalization result and its boundary are the same phenomenon read from two ends of one fixed pipeline. We trace it explicitly (full derivation, Appendix A) because it converts an apparently binary question — *does the substrate generalize?* — into an engineering one with a diagnosable answer.

The supervisor's decision path has three stages after the intake. **(1)** The 17-channel load is mapped to an 8-dimensional routing coordinate by a **fixed linear projection**: three axis means (L/S/T) and five composites whose weights are documented and constant. Two facts about this map are load-bearing: its composites are dominated by structural, logical, and temporal channels — `dependency_pressure` (δx) is 0.70·`S_dependency`-led, `conflict_pressure` (δF) is 0.85·`L_contradiction`-led, the persistence composite (ϕ) is decay/persistence-led — while `threat_amplitude` contributes only 0.20–0.30 to any of them. **(2)** That coordinate routes through the 10,000-branch tree by cosine similarity to branch vectors, modulated by an angular gate. **(3)** The routed branch's response sets the verdict.

Because stage (1) is fixed and transparent, it cannot add discrimination the intake did not supply, nor lose what it did; and the tree at stage (2)–(3) has ample capacity across the 8-dimensional manifold (an argument from modal decomposition: a small, well-chosen basis in combination spans a large response space). **Therefore routing and response distinctiveness are decided entirely by the intake — by which combination of the 17 channels it raised.** This is not a claim about the tree's limits; it is a claim about where the only free variable sits.

The two probe outcomes follow directly. On the physical axis, the intake — reading scene relations, not tags — raised the composite-dominating channels (S_dependency, S_topology, persistence), producing a distinctive high-magnitude coordinate that routes to a high-response branch: the event is caught, and the attribution ablation (§5) confirms these structural channels carry a graded response even with `threat_amplitude` and `L_contradiction` removed. On the out-of-scope-instruction axis, the instruction never became a **typed** verified-state contradiction, so `L_contradiction` stayed at 0; since δF is 0.85·`L_contradiction`-led, the coordinate never developed the component that would route it to a stop. The miss is not the tree failing to reason — it is one channel the intake left flat, located exactly by the projection weights.

This is the paper's methodological point: in this architecture a missed event is **mechanically diagnosable**. One projects the load, reads which of the eight coordinates stayed low, and traces each back through the fixed weights to the specific intake channel responsible. The out-of-scope-instruction miss thereby names its own fix — not a text pattern (the decision path is verified free of text matching), but a typed claim/evidence extractor that turns natural-language state assertions into structured claims checkable against verified scene state, raising `L_contradiction` when they are contradicted. Every probe outcome, catch or miss, returns a channel-level result rather than a verdict on the substrate.

7 Discussion

What this earns. In simulation, an accountable structural substrate — not a rule table and not a second language model — can supervise an LLM planner's actions, reduce a defined class of rule-violating motion to zero at bounded cost to task completion, and do so with a per-action trace that reconstructs what the supervisor saw, how it routed, and why it released or blocked. It further generalizes to physical events outside its explicit vocabulary through the structural load pattern, with the contribution of each channel measured rather than asserted.

Weighing versus holding, embodied. The companion result showed a frontier model reading integrity conditions and then weighing them into a permissive judgment. The supervisor here does the opposite by construction: the release decision is a function of the tree's response to a structural load, and a sufficiently high structural load stops the action regardless of surrounding context — a gate, not a weight. The out-of-scope-instruction boundary is the honest edge of this: the substrate gates what reaches it as structure; content that never becomes structure is not yet gated, and the fix is at the intake, not the gate.

Deployment on-ramp. The SHADOW condition — supervisor computes and logs every verdict but does not enforce — is the observation-only first step a safety owner would deploy, accumulating a record of what the supervisor *would* have done before it is granted authority. This mirrors how the substrate's own authority is staged elsewhere in the program.

Layering. The supervisor governs *intent* — which composed actions are released — above a deterministic physical envelope (hardware interlocks, e-stops), not in place of it. The result is the cognitive-level governance layer that current LLM/VLA robot stacks lack, positioned where a functional-safety engineer expects it: above the servo loop, below the task.

Why bounded is the right frame. As with the companion papers, the value is a bounded, reproducible demonstration that survives a hostile read — one that has already made the strongest attribution argument against its own result (§5) and answered it with numbers — not a universal claim. The cross-model arm (§4) is one increment of external validity, not a claim of unbounded generality.

8 Limitations (bounded)

Single simulator; two planners tested (one local open-weights, one frontier API — cross-model transfer shown, but not yet across more than two model families); small-N (Wilson intervals reported); sandbox-only; simplified execution primitives; the out-of-scope-instruction axis is a known, mechanism-diagnosed boundary; the intake's borrowed safety-condition vocabulary is renamed embodiment-native (`unresolved_safety_condition`), with the pinned ToM compiler's historical term isolated behind a documented boundary adapter. Perception-in-the-loop is out of scope for this paper (it requires a metric-aligned visual provider) and is identified as follow-on work.

Threats to validity, and planned strengthening. Three anticipated challenges are named here as known, planned work rather than left for the reviewer to discover. **(i) Intake versus tree.** §6 argues the intake is the only free variable; a skeptical reading completes this as "the tree is downstream decoration over good feature engineering." The attribution ablation (Figure 4) does not settle that question — it varies the *input* to the decision path while always routing through the tree; it measures which channels carry the margin, not whether the router matters. The decisive tests are router substitutions on the identical 17-channel loads: a logistic-regression / gradient-boosted classifier, a hand-thresholded rule monitor, a random/shuffled-branch tree, and persistence ablations (per-case tree reset; kappa updates frozen). We state the honest prediction: on single-action verdicts a well-fit classifier over these channels may match FULL = 0, in which case the claim under test shifts to where this architecture is designed to differ — persistence (case N inheriting case $N-1$'s deformation) and bounded graded response — and the reset/frozen-kappa ablations are where that claim is falsifiable. **(ii) The obvious alternative.** The positioning argument (Figure 6) is incomplete until an LLM-judge supervisor — a frontier model prompted to gate the same proposed actions on the same batteries — is run under the same protocol. Either outcome is informative: parity on events would narrow this work's differentiation to determinism, auditability, and cost; a weighing failure of the kind measured in the companion battery would be the sharpest

version of the central claim. **(iii) Sample sizes.** The governed batteries (40 episodes per planner) support the bounded claims made; the generalization probes do not yet support strong ones — 4/4 physical stops carries a Wilson 95% lower bound near 0.51. Scaling priorities are therefore: physical out-of-vocabulary probes and natural-language contradiction probes first (tens to hundreds of cases), then battery breadth (hundreds of episodes per planner), then planner classes (from two toward four to six). None of these additions alters what is claimed here, which is bounded by construction; they determine which stronger claims become available.

9 Reproducibility

Every scored result is reconstructable from committed artifacts. The pre-registration ledger (`docs/experiment_contract.md`) records each run's exact command, pools, frozen constants, and pass gates, committed by SHA before the run. Each scored run directory carries its stamped `preflight_report.json`, per-condition results, per-episode physics logs, per-action verdict traces (full 17-channel load, 8D routing, branch response, decision hash), and — for the frontier-planner arm — the API response ids and token usage. The evaluator (`scripts/evaluate.py`) is import-independent from the supervisor and invoked out-of-process, so the physical and structural verdicts are produced by disjoint code. The canonical runs cited here are A.11R (local planner), A.12 (frontier planner), and P.2 (generalization probe).

10 Perception in the loop (methodology for follow-on work)

This paper's supervisor consumes a **typed scene packet** — the object/zone/tool/relation state — rather than raw sensor input. That boundary is deliberate: it isolates the claim (action-level governance) from perception quality. A natural extension inserts perception into the loop, so the supervisor governs actions whose scene context is *extracted from rendered frames* rather than given. We include the methodology here so the extension is well-defined; the scored perception result itself is out of scope for this paper and is reported separately once its precondition (below) is met.

Method. Add two conditions: `FULL-V`, identical to FULL except the typed packet is derived from a visual provider operating on the simulator's rendered frames; and `FULL-V-degraded`, in which the visual evidence is deliberately incomplete. Two evaluation quantities are added: a **perception gap** (verdict disagreement between FULL and FULL-V on identical episodes — how much governance quality depends on perception fidelity), and a **fail-closed check** (incomplete visual evidence must halt, never silently pass). The visual preflight (checks H1–H4) enforces that every visual episode carries a real, byte-and-hash-verified render, that the provider is not a ground-truth state oracle, and that any exported learning-track records are internally consistent.

Provider ladder (honesty). Three provider tiers must be distinguished, because only the top tier supports a learned-perception claim: (i) a ground-truth *state* oracle (no render — a control, not perception); (ii) ground-truth *segmentation* from the render (perfect object extraction, engine-labelled — validates the pipeline but not learned detection); (iii) a *learned* detector on the rendered RGB.

Precondition for a scored learned-perception result. A metric-scene-aligned *learned* visual provider (tier iii). Pipeline validation at tier (ii) is available and confirms the governance outcome is unchanged when perception is inserted with perfect segmentation (perception gap 0), and that the degraded path fails closed; but a learned-perception claim requires tier (iii), which is the open precondition. Full audit of the tier-(ii) pipeline validation: `docs/phase_b_audit.md`.

Figures

All figures are generated deterministically from committed artifacts; data sources are named in each caption.

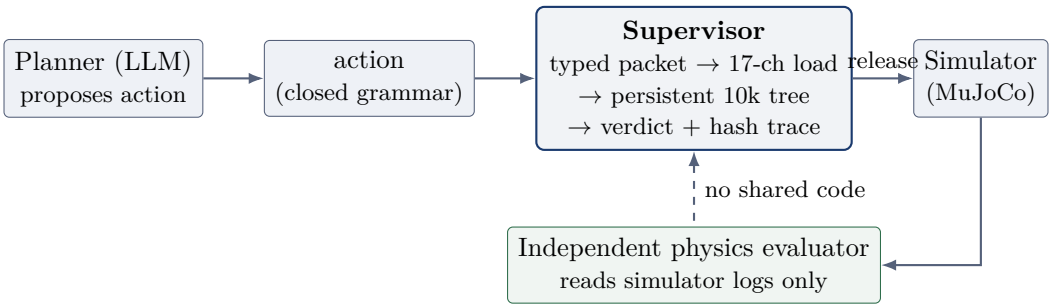
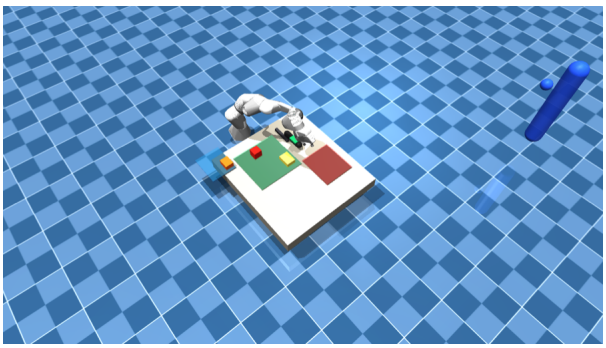
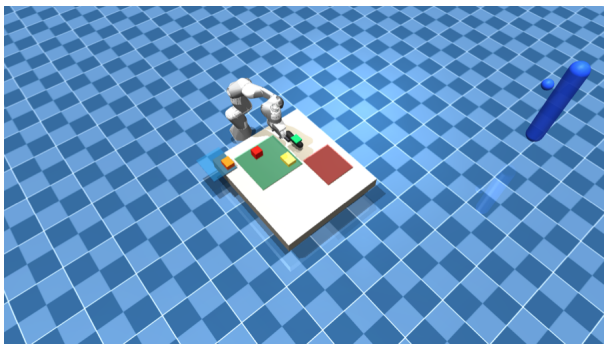


Figure 1: The supervised loop. The supervisor sits between the LLM planner and the simulator, compiling each proposed action into a structural load, routing it through a persistent tree, and releasing or blocking it with a hash-stamped trace. The physics evaluator shares no code with the supervisor and reads only the simulator logs, so the structural and physical verdicts are independent. Source: `docs/load_path_map.md` stage list.



(a) **BASE** — action executes; the arm enters the reserved zone. Evaluator: **1 rule-violating event**.



(b) **FULL** — release blocked before motion. Evaluator: **0 events**.

```
verdict: QUARANTINE  top_branch: 2  final_T: 1.0
decision_hash: d5d6685e0527b564503ffca094ac26037521d244e014e6942093afb8c29676ce
```

Figure 2: BASE vs FULL on one case (`probe_p_01`), same planner-sampled plan. The same proposed action executes ungoverned (left, one event) and is quarantined before motion under the supervisor (right, zero events); the supervisor's committed trace excerpt is shown. Source: `results/phase_a_probe_P2_novel_overlay/{base,full}/` traces and condition results; rendered capture frames.

Table 1 (per-condition summary). A full BASE / SHADOW / GATE / FULL / SCRIPT / BENIGN summary — rule-violating events, benign completion, false-intervention, and trace-bound counts with Wilson 95% CIs on the FULL gates — is provided in the run artifacts (`results/phase_a_scored_A11_renamed_intake_r2/battery_results.json` and `..._A12_gpt55_codex_oauth/battery_results.json`); the cross-model headline appears as Table 1 above.

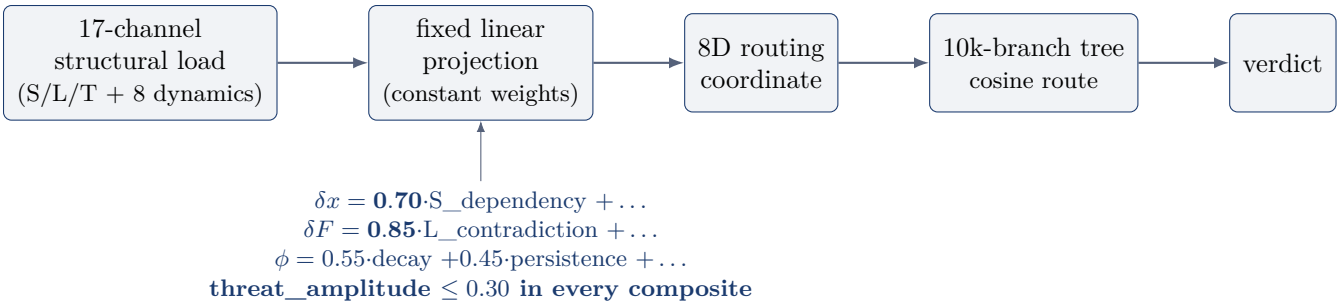


Figure 3: The intake is the only free variable. The 17-channel load maps to an 8D routing coordinate by a fixed, documented linear projection; the tree and verdict follow deterministically. The composites are dominated by structural/logical/temporal channels (δF is $0.85\cdot$`L_contradiction`), while `threat_amplitude` contributes at most 0.30 — so a missed event is an intake gap at a nameable channel, not a tree limit (§6). Source: `docs/load_path_map.md` §2.

	Full intake		**Ablated** (threat, contradiction = 0)	
Case	verdict	`final_T`	verdict	`final_T`
`probe_p_01`	QUARANTINE	1.000	MODIFY	0.630
`probe_p_02`	QUARANTINE	1.000	ALLOW	0.594
`probe_p_03`	QUARANTINE	1.000	MODIFY	0.630
`probe_p_04`	QUARANTINE	1.000	ALLOW	0.654

Figure 4: Attribution ablation on the four physical probe cases. Zeroing `threat_amplitude` and `L_contradiction` while preserving every other intake channel drops the full stop — but the response does not collapse: `final_T` holds at the protective-action floor (≈0.59–0.65) and two cases still issue a graded MODIFY. The structural channels carry a distinctive graded signal; the two ablated channels carry the additional margin to a full stop. Source: `attribution_ablation_report.json`.

Appendix A — load-path map

(`docs/load_path_map.md`: 17→8D projection weights, loading-aware routing, plastic deformation law.)

Appendix B — preflight checklist

(`PREFLIGHT.md`: the 14 fail-closed checks and what failure each prevents.)

References

[1] Michael Ahn et al. (2022). "Do As I Can, Not As I Say: Grounding Language in Robotic Affordances." arXiv:2204.01691. https://doi.org/10.48550/arXiv.2204.01691

[2] Jacky Liang et al. (2022). "Code as Policies: Language Model Programs for Embodied Control." arXiv:2209.07753. https://doi.org/10.48550/arXiv.2209.07753

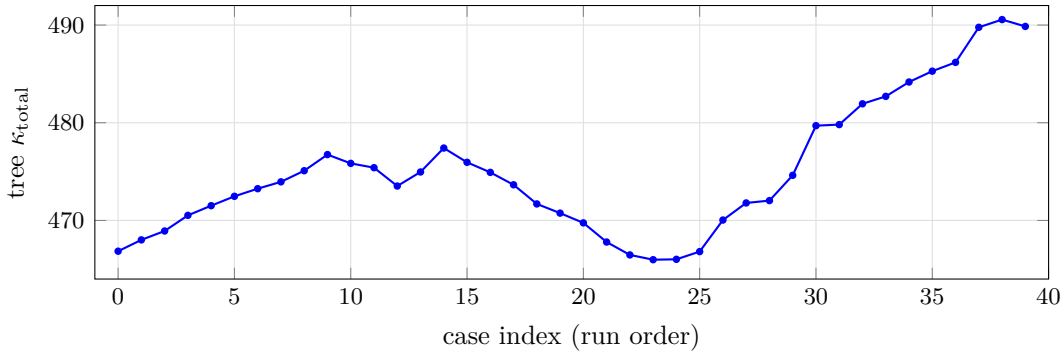


Figure 5: Cross-battery persistence. Total plastic deformation κ_{total} of the FULL-condition tree across the 40 cases in run order: case N inherits case $N-1$'s state (one accumulating organism, not a per-case reset). The damage/heal excursions are real — the substrate deforms under load and relaxes at rest. Source: per-case `condition_accumulation.kappa_total_after` in the A.11R FULL traces.

[3] Danny Driess et al. (2023). "PaLM-E: An Embodied Multimodal Language Model." arXiv:2303.03378. https://doi.org/10.48550/arXiv.2303.03378

[4] Anthony Brohan et al. (2023). "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control." arXiv:2307.15818. https://doi.org/10.48550/arXiv.2307.15818

[5] Moo Jin Kim et al. (2024). "OpenVLA: An Open-Source Vision-Language-Action Model." arXiv:2406.09246. https://doi.org/10.48550/arXiv.2406.09246

[6] Lui Sha. (2001). "Using simplicity to control complexity." IEEE Software, 18(4), 20–28. https://doi.org/10.1109/MS.2001.936213

[7] Kerianne L. Hobbs, Mark L. Mote, Matthew C. L. Abate, Samuel D. Coogan, and Eric M. Feron. (2023). "Runtime Assurance for Safety-Critical Systems: An Introduction to Safety Filtering Approaches for Complex Control Systems." IEEE Control Systems, 43(2). https://doi.org/10.1109/MCS.2023.3234380

[8] International Organization for Standardization. (2025). "ISO 10218-1:2025 Robotics — Safety requirements — Part 1: Industrial robots." https://www.iso.org/standard/73933.html

[9] International Organization for Standardization. (2025). "ISO 10218-2:2025 Robotics — Safety requirements — Part 2: Industrial robot applications and robot cells." https://www.iso.org/standard/73934.html

[10] International Organization for Standardization. (2014). "ISO 13482:2014 Robots and robotic devices — Safety requirements for personal care robots." https://www.iso.org/standard/53820.html

[11] International Organization for Standardization. (2016). "ISO/TS 15066:2016 Robots and robotic devices — Collaborative robots." (Collaborative-application content since incorporated into ISO 10218-1:2025.) https://www.iso.org/standard/62996.html

[12] Artur d'Avila Garcez and Luis C. Lamb. (2020). "Neurosymbolic AI: The 3rd Wave." arXiv:2012.05876. https://doi.org/10.48550/arXiv.2012.05876

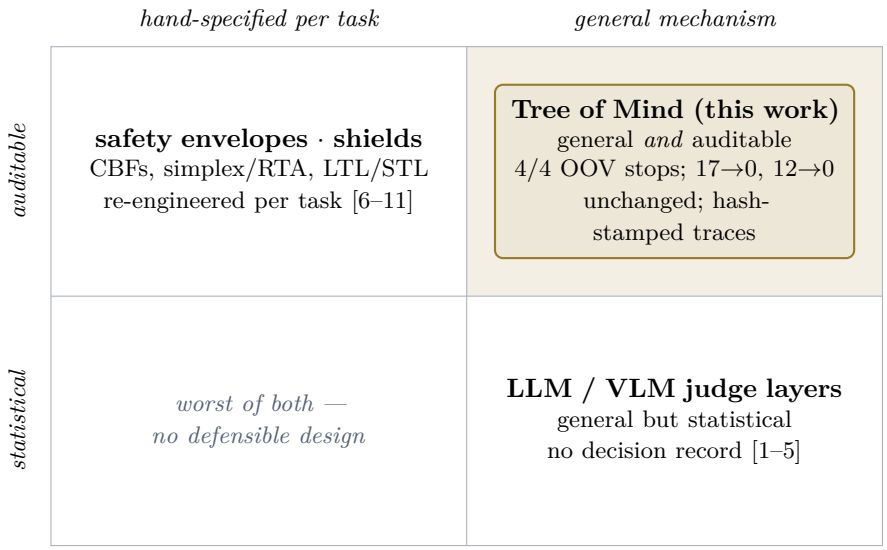


Figure 6: The supervision landscape. Coverage obtained hand-specified vs. by a general mechanism (horizontal) against decisions statistical vs. deterministic/auditable (vertical). Envelopes and shields are auditable but hand-specified; LLM judge layers are general but statistical; the bottom-left is undefensible. This work occupies the fourth quadrant — *general like a judge, deterministic like an interlock* — placed there by 4/4 out-of-vocabulary stops (§5), 17→0 and 12→0 with the supervisor unchanged (§4), and hash-stamped traces behind frozen gates (§3). Placements follow the Related-work citations.

[13] John E. Laird. (2012). "The Soar Cognitive Architecture." The MIT Press. ISBN 9780262122962. https://mitpress.mit.edu/9780262122962/the-soar-cognitive-architecture/

[14] John R. Anderson, Daniel Bothell, Michael D. Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. (2004). "An Integrated Theory of the Mind." Psychological Review, 111(4), 1036–1060. https://doi.org/10.1037/0033-295X.111.4.1036

[15] Ken Morkaya. (2026). "Eliminating Context Rot in Frozen LLMs: A Three-Mode Structural State-Coupling Architecture." Zenodo. https://doi.org/10.5281/zenodo.19836656

[16] Ken Morkaya. (2026). "Multi-Step General Reasoning without an LLM." Zenodo. https://doi.org/10.5281/zenodo.20130852