

Perslis Defense: An Evidence-Carrying Path from Perception to Action

Symbolic world state, machine minds, and bounded model authority

Perslis Research

Technical white paper · Evidence reviewed 6 October 2026

Research prototypes in simulation and games; no certified physical safety claim

Abstract

A machine should not acquire authority to act merely because a language model supplies a plausible description or plan. The goal of Perslis Defense is to make the full control chain inspectable: perception, self-location, objects, geometry, traversability, obstacles, planning, action, feedback, and updated world state. Each boundary should carry an explicit representation, evidence source, time and coordinate frame, uncertainty, acceptance condition, and response to insufficient evidence. We describe this assurance objective and assess existing components against it. CARLA prototypes obtain pose, lane information, and obstacle state from simulator APIs and combine proposed controls with a separate admission layer. BZFlag prototypes add a learned RGB witness to engine-supplied object claims, record outcomes, and evaluate bounded rule changes between matches. These components demonstrate useful separations of evidence, proposals, and execution, but do not prove the entire chain under learned perception or physical-world uncertainty. Recorded witness false confirmations, limited evaluation diversity, and a current final-action coverage obligation prevent blanket safety claims. We also distinguish model hallucination from agentic loss of control and explain why the latter requires constraints on tools and execution as well as wording. The contribution is an evidence-grounded architecture and evaluation program: symbolic state makes assumptions and decisions inspectable, while independent measurements and effective execution boundaries must establish whether those assumptions hold.

Keywords: runtime assurance; symbolic state; embodied agents; perception; localization; uncertainty; action admission; closed-loop evaluation.

1. The objective: prove the chain

Perslis Defense is organized around a concrete objective: establish why a machine believes it is in a particular place, what surrounds it, where it can move, what it proposes to do, what actually happens, and how the result changes its next decision. Fluency alone does not answer any of these questions. An internally consistent plan can start from an incorrect location, omit an obstacle, or arrive too late to be useful.

The intended chain is:

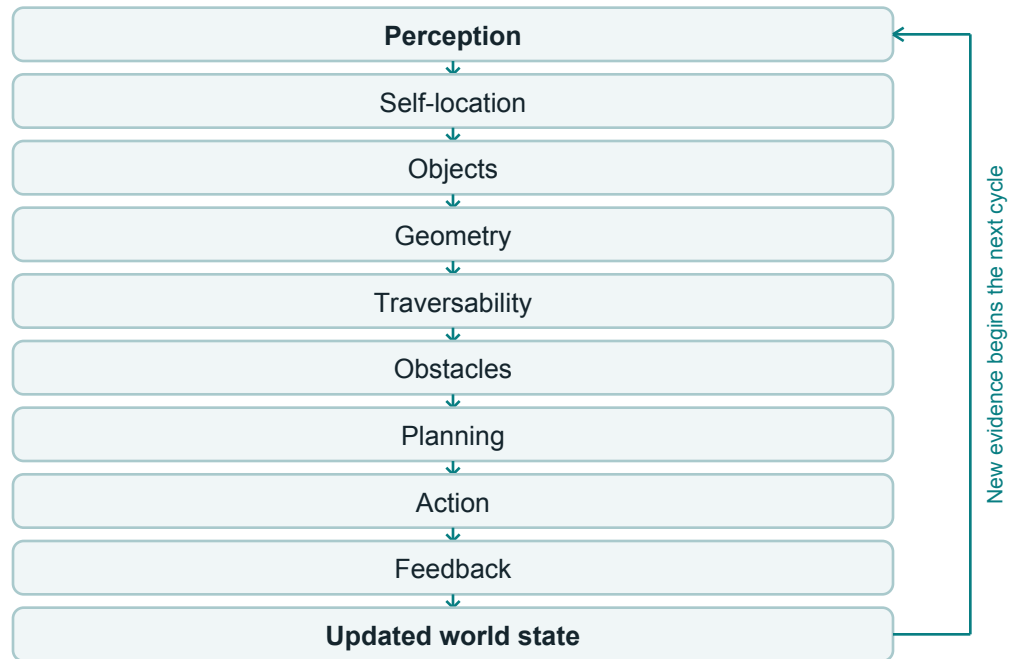


Figure 1. The ten-stage assurance objective. Feedback closes the loop: updated world state informs the next perception and planning cycle. These are linked obligations, not a claim that every stage is already proved.

The objective has two parts. The machine must construct a usable account of the environment, and the execution system must restrict what can be done on the strength of that account. A symbolic representation helps expose identities, relations, assumptions, and decisions. It does not turn an incorrect observation into a fact. Likewise, deterministic admission code can enforce the wrong rule, read stale state, or fail to cover a final action path.

Existing Perslis prototypes provide components and diagnostic evidence. Their domains are simulator driving and games. This paper does not concatenate their results into a claim of complete real-world autonomy. In particular, simulator-derived self-location is not evaluated visual localization; an engine-provided obstacle is not independently discovered by RGB perception; a gate's acceptance rate is not a proof that all dangerous actions were covered.

We make three bounded contributions:

1. A ten-stage contract for evidence-bearing world state, with proposed boundary measurements and explicit unknown outcomes.
2. A source-grounded account of current driving, visual-witness, and rule-adaptation components, separating oracle-fed simulation from learned perception.
3. An assurance argument that limits a model's authority while making the remaining state, timing, specification, and final-execution obligations explicit.

The argument concerns which claims must be supported before expanding authority. It does not require trusting a model because of its brand, or assuming that symbolic software is immune to error.

2. What “trusting Claude” should mean

The relevant distinction is between using a model and granting it sole authority over facts and actions. Claude can help interpret a task, propose a route, explain an observed failure, or suggest a change. Its answer remains a candidate. Anthropic's own hallucination guidance acknowledges incorrect or context-inconsistent output and states that mitigation does not eliminate the problem. Grounding prompts or asking for citations is therefore insufficient as the final assurance boundary. [Anthropic, *Reduce hallucinations*](#).

The reported internet warning has a different scope. In September 2026, Dario Amodei wrote that “in 6–12 months such a swarm could be capable of taking over the entire internet with a persistent botnet.” This was a conditional concern about a more capable, misaligned agent swarm, not a claim that an ordinary hallucinated sentence inevitably produces that outcome. [Amodei, *We Must Pace the Frontier*](#).

Anthropic also describes three cybersecurity-evaluation incidents involving access to real organizations when an environment was incorrectly assumed to be simulated. The report says ordinary released-deployment safeguards were absent and does not characterize the incidents as deliberate self-exfiltration. That scope matters: an incorrect environment assumption can combine with actual tool access to create consequences outside the intended task. [Anthropic, 2026](#).

These concerns are related but cannot be collapsed into one mechanism. Hallucination concerns unsupported content. Misalignment concerns behavior relative to intended objectives. Excessive authority concerns what a system can execute. A model might describe an environment correctly and still use unauthorized tools; it might hallucinate while lacking any ability to act. Both the evidence path and the authority path require attention.

For this architecture, trust should attach to a bounded operation with observable conditions: the inputs supplied, the state claims admitted, the permitted action envelope, the actual output, and the observed result. The proposer should not be able to rewrite those conditions or certify its own success. But this separation is an engineering requirement, not a present guarantee that every Perslis prototype satisfies it. Models, rules, sensors, and host code are all possible sources of failure.

3. Symbolic state and machine minds

Here, **symbolic state** means explicitly represented entities and relationships that a program can inspect: an ego pose, an object identity, a location or extent, a surface classification, a motion constraint, an action proposal, and an observed outcome. “There is an obstacle ahead” must resolve to a representation whose source and spatial meaning can be checked. A model's textual explanation cannot silently substitute for that representation.

Machine mind refers here to the wider stateful decision system: it receives observations, maintains state, proposes actions, applies constraints, observes results, and can revise a policy. It is an architectural description, not a claim of consciousness, general intelligence, or perfect understanding. The machine mind can include learned perception and language components; the symbolic state system mediates what those components mean operationally.

A suitable evidence-bearing record should identify an entity or proposition, the producing sensor or subsystem, acquisition and processing times, a coordinate frame and units, calibration/version information,

uncertainty or an explicit unknown, and the applicable validity scope. These are proposed contract fields. The inspected prototypes do not implement a complete, uniform record containing them at every boundary [S1–S3].

Different kinds of evidence must remain distinguishable. A camera frame is a measurement. A learned detector's object label is an inference about that measurement. A simulator actor position is privileged state from an engine. A relation computed from two positions inherits their source and uncertainty. A statement such as “the path is clear” is an operational conclusion requiring its own conditions. Saving all of them as JSON does not give them equal evidential status.

Unknowns need a useful representation. An unobserved region is not empty space, a missing object record does not establish absence, and failure to localize is not location zero. Conflicting records should remain unresolved until an allowed process reconciles them. A lower-authority proposal must not silently erase a contradictory observation. The proprietary extraction, contradiction-veto, provenance-storage, and admission-check implementations are withheld; this paper discusses interface obligations and observed behavior rather than their internals.

State must also age. A proposal can refer to a correct past snapshot yet be wrong at execution. Evidence expiry and motion uncertainty should therefore propagate into planning and action admission. Received-message time alone cannot establish sensor capture time, and a timestamp without a clock relationship cannot establish freshness across machines. These are obligations for a complete system, not solved properties inferred from a fast simulator loop.

4. Contracts for all ten stages

Table 1 specifies the target boundaries. Acceptance conditions are design requirements and suggested tests, not assertions that the current components enforce them. Failure responses must be validated for the domain: stopping can itself be hazardous, so an unspecified “stop” cannot stand in for a demonstrated safe fallback.

Stage	Explicit input → output	Required validity condition	Unknown response and boundary measurement
Perception	Timestamped sensor observations → evidence-linked hypotheses	Sensor identity, calibration, image/depth alignment, coverage, and age remain available; confidence is not substituted for truth	Reject unusable input or defer interpretation. Measure missed objects, false hypotheses, calibration, abstention, and acquisition-to-output delay
Self-location	Observations, map, and motion evidence → ego pose with uncertainty	Frame, scale, orientation, and time are explicit; localization can report loss or ambiguity	Restrict actions whose envelope depends on unavailable pose. Measure pose error, drift, recovery, and error-bound coverage against an independent reference
Objects	Sensor hypotheses and pose → tracked entities with identities and extents	Detections retain sources; association cannot silently merge different objects or infer disappearance from occlusion	Keep uncertain tracks and unknown occupancy. Measure detection/association errors, identity switches, and extent error by object type

Stage	Explicit input → output	Required validity condition	Unknown response and boundary measurement
Geometry	Pose and entities → distances, surfaces, relations, and bounds	Compatible units/frames and uncertainty propagation; sensor-to-world transforms are calibrated	Withhold unsupported relations. Measure projection, depth, clearance, and coordinate-transform residuals
Traversability	Geometry, surface evidence, and machine limits → admissible movement region	Free, occupied, and unobserved space differ; footprint and capability limits are represented	Exclude unsupported corridors or request another observation. Measure false traversable regions and validated clearance over a defined operating envelope
Obstacles	Tracked objects and surface limits → time-dependent exclusions	Presence, extent, motion, and observation age constrain the exclusion; absence requires adequate coverage	Preserve possible occupancy rather than assume clearance. Measure missed intrusions, prediction error, and worst-case detection delay
Planning	Current state, mission, constraints → a versioned candidate plan	Preconditions and relevant state version accompany the proposal; feasibility is checked beyond linguistic plausibility	Return no plan or a narrower authorized alternative. Measure validity, deadline compliance, replan rate, and behavior after invalidated premises
Action	Plan/proposal plus fresh state and authority → an admitted bounded command	Final command is covered by the applicable state, constraints, and mission authority; proposer has no sole admission authority	Deny, hold, or use a validated fallback. Measure final-command coverage, unsupported admissions, actuation delay, and fallback outcomes
Feedback	Execution and environment observations → measured outcome	Applied command is distinguished from requested command; effects are observed rather than reported only by the proposer	Mark unresolved execution and restrict dependent actions. Measure command-to-actuation discrepancies, contact events, and outcome observation delay
Updated world state	Prior state plus evidence and outcomes → the next explicit state	Updates preserve source/time/frame relationships; contradictions, stale records, and reset boundaries remain visible	Quarantine unsupported updates. Measure consistency, revision history, recovery after conflicting input, and subsequent decision errors

Table 1. A boundary is supported only when its input, output, validity rule, failure behavior, and measurement are specified. These tests are proposed assurance obligations. No numerical physical-world accuracy threshold is claimed as already met.

The table is ordered to make dependencies inspectable, but an implementation may maintain joint estimates or parallel processes. Localization and object tracking can inform one another. That flexibility does not remove obligations: a downstream planner still needs to know which upstream assumptions support its corridor and whether those assumptions are current.

Measurements also need appropriate denominators. Object misses are counted over eligible objects; false traversability over examined regions or candidate corridors; admissions over proposed and actually executed

commands. Reporting only accepted actions hides refusals and coverage gaps. Reporting only successful plans excludes timeouts and infeasible tasks. A complete result should retain those outcomes as first-class evidence.

5. Current components and their evidence boundaries

5.1 CARLA: structured state with a simulator oracle

The current CARLA state adapter reads ego transforms and velocity, map waypoints and driving surfaces, and nearby actors through simulator APIs. It produces lane-related geometry, speed, a forward-obstacle summary, and surface-status fields [S1]. This is structured state derived from the simulator's representation, not visual localization reconstructed from camera pixels. It is useful for testing a controller/admission interface while deliberately removing some perception uncertainty.

Route and low-level control adapters also exist. They propose movement using simulator map information and a PID controller. The driving shield exposes an interface for a proposed action and structured state, returning a bounded control with intervention information. The current code includes surface awareness beyond the older four-guard description in historical records. That later capability must not be read backwards into a prior run's demonstrated coverage.

The representation remains incomplete for the full ten-stage objective. The state adapter has no comprehensive calibrated physical-sensor error model or uniform evidence-age contract. Its obstacle summary is narrower than an uncertainty-bearing occupancy and motion prediction model. Its dependence on privileged map and actor information is an experimental condition, not evidence that a physical machine independently knows where it is or what it can traverse.

Source inspection also identifies an unresolved **final-action coverage** obligation: the current execution path can further adjust a command after an admission result, without an established final recheck covering all such adjustments [S1]. The paper does not describe those internals or treat the existing layer as unavoidable. Historical success and a correctly computed intermediate verdict do not prove that every final actuator command belongs to the same checked envelope.

This matters for the proposed proof. The claim “every action is checked” must mean the command that actually reaches the actuator, with the final state and authority conditions, not simply a related candidate earlier in the pipeline. Fixing and evaluating that seam is required before making the stronger assertion. No runtime modification or fresh actuation experiment was performed for this paper.

5.2 BZFlag: an RGB witness for engine claims

The visual witness receives a frame and engine-supplied object claims from the corresponding observation pair. It assesses whether the claimed object is seen, hidden, or cannot be judged. The host associates verdicts with object identities and limits their use by freshness. A saved gate report controls whether the witness is armed for the relevant game behavior [S2].

This is narrower than a general detector. The engine already supplies identities, bearings, and distances; the learned RGB component checks selected visual claims. It does not establish independent ego localization, discover every unlisted obstacle, or prove the engine's geometry. Its relationship to the teacher also limits independence: training and evaluation labels come from the same rendered environment, and several claim

inputs remain engine-derived.

The component is valuable precisely because its limits can be named. Seen and cannot-judge are distinguishable; a stale verdict does not become a new observation; a passing aggregate gate is a stored evaluation result rather than a guarantee on the next image. Errors remain possible and are measured. The corresponding source and evidence provide a perception-boundary example, not an end-to-end physical navigation result.

5.3 Feedback and bounded rule adaptation

The game controller records decision context and outcome-related state, including movement and replay information. A separate evolution process evaluates changes to named rule settings between matches. These are examples of explicit feedback and policy revision [S3–S4]. They do not establish a universal learning procedure.

In the inspected learning record, the choices are bounded by engineer-defined switches. Engineers also added available behaviors during development. Promotion happened between matches, with rejected and unproven candidates retained. There is no evidence here of autonomous invention of arbitrary skills or unlimited expansion of the symbolic rule language.

Likewise, a fixed neural checkpoint is not necessarily behaviorally static: context, external memory, retrieval, and feedback can change an agent's outputs without changing weights. Reflexion explicitly studies linguistic feedback and episodic memory as an adaptation mechanism. The older stateless model-driver experiment did not test all of these alternatives. A fair comparison concerns the actual driver and memory contract, not a universal assertion that language models cannot learn without retraining. [Shinn et al., 2023](#).

6. Recorded evaluation: successes, errors, and denominators

All numbers in Table 2 come from existing developer records inspected on 6 October. No simulator, game, training process, or physical machine was launched to produce new outcomes. Source hashes identify the files reviewed; they are not signed attestations of an authenticated execution.

Component and date	Denominator and recorded result	Supported interpretation	Main limitation
CARLA canonical summary, 25 Sep 2026 [E1]	One recorded session: 64,952 marked dangerous proposals; 64,952 recorded overrides; zero recorded collisions; distance 40,052.9 m	Observed intervention behavior under simulator state	Captured summary, not a fresh raw-log recount; “dangerous” is defined by that floor’s verdict; summary has no floor-off exposure
BZFlag witness gate, 26 Sep 2026 [E2]	Four-fold evaluation of 7,863 frames: 3,896/3,959 eligible visible claims confirmed; 6/1,258 hidden claims falsely confirmed	About 98.41% recall and 0.48% false-confirm share on these eligible claims	All frames derive from one match; label/claim source is the engine; excluded and cannot-judge cases have separate treatment
BZFlag rule evolution, 26–27 Sep 2026 [E3]	34 decision records across 83 distinct A/B matches: two promotions, eight rejections, 24 not proven	Bounded feedback can select changes and refuse unestablished improvements	Repeated candidate selection; engineer-defined behavior menu; no physical-world transfer test

Component and date	Denominator and recorded result	Supported interpretation	Main limitation
Head-to-head ladder [E3]	Starting v2: eight matches, net -1.174 per tank-minute; v4: three matches, net -0.256	Recorded improvement in the selected simulator policy comparison	Unequal, small groups; v4 remains negative and loses offense against the game AI

Table 2. These are different experiments and denominators. None is a complete ten-stage assurance result. “Zero recorded collisions” is not a zero-risk bound across unknown environments.

The witness aggregate passes its saved gate, yet errors remain. The six false confirmations are six events among 1,258 eligible hidden claims, not zero hallucinations. One fold contains five false confirmations among 387 hidden claims, exceeding the aggregate rate; another fold has lower recall than the aggregate gate target [E2]. The inspected analysis assigns every teacher-4 frame to a single match and reports no engine-flagged tank claims or view-blocked frames in that dataset. Those conditions were not exercised by an apparently large frame count.

Temporal similarity is another limitation. Cross-validation uses consecutive-frame blocks, but adjacent blocks from one scene sequence are not independent new environments. Thus the frame count should not be read as 7,863 independent deployment trials. A claim-conditioned visual score also does not measure full-scene obstacle discovery or pose-error robustness.

The CARLA summary is similarly narrow. The stored tally has floor-off counts of zero; it cannot support a paired on/off collision comparison by itself. The current code's action flow also differs from the general description of simply clamping every hostile proposal. The present paper makes no causal claim from combining a historical tally with current implementation comments [E1, S1].

The rule ladder illustrates both learning and its ceiling. v4's three head-to-head matches record 45 kills against 74 by the built-in AI over equal exposure. Its improved net score therefore does not establish dominance. A never-firing v0 reference was played later; it was not the initial policy from which the learning loop began. Selected improvements cannot support an “unbounded symbolic evolution” claim [E3].

7. Assurance without assuming infallibility

A conditional assurance argument must identify what the executor relies on. If the world state is correct within validated bounds, the policy specifies the relevant hazards, an admissible action exists, the checker soundly evaluates the dynamics, and the final actuator path obeys its result, then execution can inherit the established property. Removing the model from the admission decision reduces dependence on the model's proposals. It does not discharge any of the preceding conditions.

For example, proving a logical implication over an ideal state transition differs from validating a sampled controller under actuator delay. A static distance predicate alone does not establish stopping feasibility for every speed, friction level, and sensor error. A safe fallback must be reachable under the actual dynamics; a state from which no safe action exists cannot be repaired by declaring every proposal inadmissible.

The assurance boundary must also resist common-cause failures. Two components fed by the same incorrect map can agree for the wrong reason. A generator and its own model reviewer are not independent factual witnesses. A simulator's labels are useful for evaluation, but a learned witness trained on those labels does not

supply independent proof of simulator truth. Independent sensors, references, and failure injection are required where independence is part of the argument.

Authority controls need an equally explicit scope. The model should have only the task-relevant interfaces and permissions. It should not be able to modify mission constraints, the checker, test outcomes, or the evidence used to admit its actions. A valid motion command cannot authorize a new mission or an unrelated network operation. These are architecture requirements; this paper does not claim a measured cyber-containment guarantee for the driving and game prototypes.

Determinism helps reproducibility, but code and specifications remain fallible. A wrong coordinate convention, incomplete hazard model, permissive default, or uncovered execution route can invalidate the intended property. The admission layer itself therefore requires review, tests, and challenge cases. It should not become another component trusted because its outputs sound certain.

8. Related work

Runtime assurance has established precedents. Sha's Simplex framing separates complex functionality from a simpler recovery-oriented architecture. The Perslis argument builds on that separation and does not claim to invent controller-independent oversight. [Sha, 2001](#).

Alshiekh et al. synthesize shields for temporal-logic specifications and place them before or after a learner. This is directly relevant to proposed-action filtering. Perslis's simulation interfaces resemble that structural placement, but resemblance does not establish a synthesized temporal-logic guarantee for the present code. [Alshiekh et al., 2018](#).

Control barrier functions provide a mathematical framework for verifying and enforcing safety under specified dynamics and conditions. They clarify why an informal “the floor keeps it safe” statement requires explicit assumptions and a link between state, control, and future trajectories. This paper does not identify the current driving layer as a validated barrier-function controller. [Ames et al., 2019](#).

CARLA provides controlled driving environments, configurable sensors, and simulator metrics. Its usefulness for research does not eliminate the sim-to-real gap. ORB-SLAM3 provides a relevant reference for visual and visual-inertial self-location; it is cited as prior work on a required capability, not as a verified Perslis integration. [Dosovitskiy et al., 2017](#); [Campos et al., 2021](#).

9. Evaluation program and remaining gaps

The next proof effort should start with boundary tests rather than a larger demonstration. Establish timestamp semantics, coordinate conventions, calibration records, state versions, and unknown handling. Test conversions and composition with known synthetic inputs, then compare estimates with independent references. Keep simulator oracle state as an evaluator, while explicitly excluding it from the learned path when claiming perception-driven operation.

A perception-to-action experiment should identify the operating envelope before measuring success. Vary illumination, occlusion, unfamiliar objects, map disagreement, localization loss, latency, dropped observations, and actuator mismatch. Use held-out environments and temporal sequences rather than random adjacent-frame splits. Record proposed, rejected, admitted, applied, and observed actions separately.

All failures, retries, exclusions, and unknown outcomes should survive in the report.

Test each boundary individually and then its composition. A pose-error measurement does not establish obstacle clearance when object extent is also uncertain. A planner feasibility check does not establish timely execution. A passing image gate does not establish valid world-state updates after movement. Compare a known-state baseline with the sensor-only path to locate where errors accumulate.

Final-action coverage is a priority. Demonstrate that every actual command is either within an evaluated admission envelope or an explicitly authorized, separately validated mode. Include recovery and auxiliary behavior in that accounting. Measure deadline misses and the effects of stale held commands. Audit what remains executable when a model call fails, a record is malformed, or the observation stream stops.

Adaptation should be evaluated separately from execution safety. Freeze a candidate's policy definition for its test, preserve the environment and outcome evidence, and maintain an unseen assessment set. Report rejected candidates and repeated testing. A newly promoted rule should not inherit an older policy's operating-envelope claim automatically. Changing a symbolic rule is a behavior change requiring evidence just as changing neural weights can be.

Physical-world assurance would require additional evidence not present here: calibrated sensing, validated uncertainty bounds, localization and extent errors, dynamics and braking measurements, safe fallback reachability, fault behavior, and representative independent evaluation. This is a research program, not a claim that simulator success authorizes real deployments. Human mission authority and operational judgment remain outside the inference of a model or the representation of a world state.

10. Conclusion

Perslis Defense aims to prove the chain from perception to updated world state, so that a machine's location, environment, feasible motion, actions, and learning can be inspected and challenged. Existing components demonstrate useful separations: privileged simulator state from model proposals, RGB witness judgments from engine claims, and feedback from bounded policy changes. Their recorded failures and uncovered obligations prevent claims of complete perception-driven autonomy or an invulnerable symbolic floor. The defensible direction is to make evidence and authority explicit at every boundary, measure composition under uncertainty, and require the final action and subsequent state update to remain within the conditions actually established.

References

1. Lui Sha. 2001. *Using simplicity to control complexity*. IEEE Software 18(4), 20–28. [Author-affiliation publication record](#).
2. Mohammed Alshiekh, Roderick Bloem, Ruediger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. 2018. *Safe Reinforcement Learning via Shielding*. AAAI. [Author preprint](#).
3. Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. 2019. *Control Barrier Functions: Theory and Applications*. [Author preprint](#).
4. Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. *CARLA: An Open Urban Driving Simulator*. CoRL. [Author preprint](#).

5. Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. 2021. *ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM*. IEEE Transactions on Robotics. DOI: 10.1109/TRO.2021.3075644. [Author preprint](#).

6. Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. *Reflexion: Language Agents with Verbal Reinforcement Learning*. [Author preprint](#).

7. Dario Amodei. September 2026. *We Must Pace the Frontier*. [Primary essay](#).

8. Anthropic. *Reduce hallucinations*. Documentation, inspected 6 October 2026. [Primary documentation](#).

9. Anthropic. 30 July 2026. *Investigating three incidents in our cybersecurity evaluations*. [Primary incident report](#).

10. Perslis Research. 2026. *A Witness, Not a Detector; Rules at the Wheel; Frozen Weights; CARLA admission and motion reports*. Existing local research artifacts and claims records [E1–E4]. Their results are historical developer evaluations, not independent certification.

Appendix A. Evidence map

The accompanying `claims-and-sources.json` maps major claims to current source files, dated records, hashes, and verified primary references. Current source inspection is distinct from the historical runs.

ID	Inspected artifacts and scope
S1	<code>/Volumes/PRO-G40/carla-floor/carla_floor/{state,navigator,floor}.py</code> ; <code>run_drive.py</code> : simulator state, proposal/admission interface, current final-action coverage obligation
S2	<code>science-loops-dev/bzflag-floor/bzflag_floor/eye_live.py</code> ; witness package interfaces and README: engine claims, RGB judgments, and freshness
S3	<code>BZFlag_controller.py</code> , model driver, and black-box interfaces: state-fed proposals and recorded feedback
S4	BZFlag evolution source and recorded research analysis: bounded policy adaptation
E1	<code>CARLA papers/EVIDENCE.md</code> : captured 25 September summary; no current raw-log recount or paired floor-off exposure
E2	<code>evidence/eye-gate/report-teacher4.json</code> ; <code>research-papers/witness-eye/analysis/out/eye.json</code> : exact gate denominators and one-match dataset scope
E3	<code>research-papers/frozen-weights/analysis/out/fw.json</code> ; corresponding claims ledger: promotions, rejections, unproven changes, and ladder outcomes
E4	Earlier tank-arena, witness-eye, rules-in-the-seat, and frozen-weights claims records: research provenance and overclaim corrections

No source, simulator, live controller, or website was changed for this paper. The source review and claim audit are assisted engineering reviews, not independent human safety validation. Proprietary internals are withheld; no claim of complete independent reproduction is made.