INSPECTIONBENCH: EVIDENCE-GROUNDED EVALUATION OF AGENTIC INDUSTRIAL INSPECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

With the rapid development of industrial intelligence and robot-assisted inspection, reliable physical-to-operational coupling from evidence acquisition and physical verification to downstream operational decisions has become a key bottleneck for deploying autonomous inspection and predictive-maintenance agents in real-world industrial environments. Existing benchmarks remain limited to static or single-modality observations, hand-specified scenarios, or outcome-oriented evaluation that fails to determine whether the required evidence was actually acquired. We introduce INSPECTIONBENCH, an execution-grounded benchmark for agentic physical AI in industrial inspection, comprising 4,120 canonical episodes generated from 1,552 grounded worlds and 21 scenario templates across six industrial asset types, in which agents acquire visual, operational, and telemetry evidence with explicit provenance and execute inspections before making downstream operational decisions. A world-to-scenario compiler automates matched-episode generation through parameterized control of evidence visibility, modality and tool access, temporal validity, and guidance while preserving the underlying world and gold decision, adjudicated by domain experts where needed. We evaluate five complementary capabilities across five frontier models, finding that correct-action rates exceed evidence-supported success in 20 percentage points and exact physical-constraint identification up to 37.2 points, highlighting capability gaps that final task success alone can obscure. Code and evaluation artifacts are available in Code.

1 INTRODUCTION

Autonomous inspection can reduce repetitive work, reach hazardous or hard-to-access locations, and detect degradation between human inspection rounds. Industrial monitoring, however, produces anomaly alerts, not operational decisions. An alert may indicate an equipment fault, a drifting sensor, a transient operating condition, or maintenance already in progress. Operators also receive more alerts than they can realistically review (Engineering Equipment and Materials Users' Association, 2024; International Society of Automation, 2016; Hashim et al., 2022). Dispatching an inspection for every alert consumes scarce capacity, while acting on an unverified alert risks unnecessary maintenance, missed failures, or unsafe interference with ongoing work (de Jonge & Scarf, 2020). Operators must therefore decide what evidence they need before taking action.

Mobile inspection robots can collect this evidence. Figure 1 shows a legged platform collecting gauge readings, vibration, and acoustic signatures at an industrial site. AutoInspect reports comparable deployments at a mine, a chemical plant, and nuclear facilities (Staniaszek et al., 2025). In an offshore-wind case, ANYbotics reports using robots to verify turbine alarms before sending crews; more than 70% of the alarms were false positives. This figure comes from a vendor report rather than a controlled study (ANYbotics, 2025). A practitioner who operates an inspection robot at a process plant described the same verification practice to us.

The robot itself can also fail. AutoInspect reports unreachable locations, failed actions, low-battery interruptions, and localization errors (Staniaszek et al., 2025). Recording an incomplete observation, or one from the wrong location, as successful verification corrupts the maintenance record. A blog from Quarero Robotics describes rechecking an anomaly with a second capture before escala-

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.



Figure 1: Robot-assisted industrial inspection.

tion (Nagel, 2026). An autonomous AI agent must decide what to inspect, which channel to use, and when to stop, taking both uncertain evidence and robot-access constraints into account.

Various dataset and benchmarking framework has been developed recently for evaluating AI Agents on industrial asset monitoring. Task-aware industrial datasets evaluate maintenance question answering (Shyalika et al., 2026) and failure classification (Liu et al., 2026; Ho et al., 2026), but supply the evidence instead of letting the model choose what to acquire. Active-perception benchmarks allow agents to seek additional views through embodied exploration (Hong et al., 2026; Yao et al., 2026), anomaly analysis (Miao et al., 2026), or robotic scanning (Chen et al., 2026). These episodes end in an answer, hazard report, anomaly label, or reconstruction. Industrial workflow benchmarks go further by executing actions that update enterprise state AssetOpsBench (Patel et al., 2026), PHMForge (Li et al., 2026a), but rely on digitally accessible tool state and never dispatch a robot. No benchmark scores the full sequence from an alert, through physically acquired evidence, to an enterprise action (See Table 1).

In this paper, we discuss an execution-grounded AI agent benchmark. An AI agent receives an alert and asset context, invokes MCP tools such as `navigate_to` and `read_gauge` to acquire inspection evidence, and turns the returned observations into an operational decision, COMMIT, ESCALATE, or ABORT. It then issues the matching enterprise action through tools reused from AssetOpsBench, such as `create_WO` and `list_assets`. Section 3 describes the construction. Our contributions are as follows.

- We introduce **InspectionBench**, an execution-grounded benchmark that follows agents from an industrial alert through evidence acquisition to operational action across 4,120 canonical episodes and 1,552 grounded worlds. Its 25-tool MCP interface integrates Boston Dynamics Spot SDK (Boston Dynamics, 2026) capabilities with AssetOpsBench, connecting robot inspection with downstream enterprise actions.
- We ground inspection in evidence whose origin, accessibility, and timing determine whether it can justify an action, combining 116 facility images, 1,296 rendered perception scenes, and 24,601 replayed operational records. These sources extend AssetOpsBench with annotations, asset-level provenance, robot capabilities, access constraints, and observation timing.
- We turn fixed inspection cases into controlled experiments by changing what agents can observe and do while preserving the underlying world and correct decision. A reusable compiler instantiates 21 world-independent scenario templates and 52 variation identifiers across grounded worlds, varying evidence visibility, modality, tool access, temporal validity, and guidance.

Empirical Evidence Our experimental analysis shows that correct operational decisions can mask inspection failures: four of five frontier multimodal models exhibit gaps exceeding 20 percentage points, reaching 29.8, between decision correctness and success grounded in evidence actually ac-

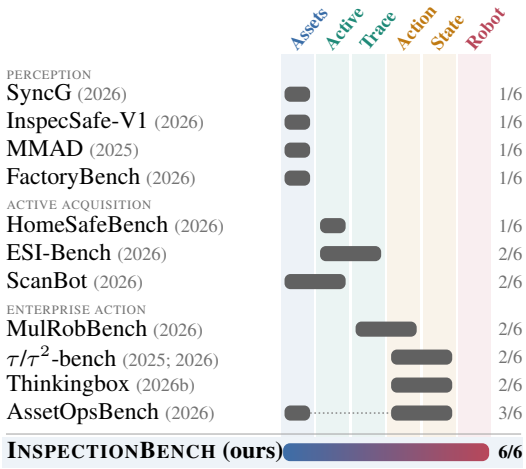
quired. Exact physical-constraint identification trails decision correctness by up to 37.2 points, while acquisition, procedural, and temporal diagnostics reveal further breakdowns between inspection and action.

2 Related Work

We review prior work on industrial evidence and perception, active acquisition, and enterprise action. Table 1 compares six capabilities. *Active* means that the agent chooses the next observation; *Trace* indicates that scoring covers the acquisition sequence as well as the answer. *State* means that an action updates a persistent enterprise record. *Robot* indicates that tool calls drive a mobile inspection robot, rather than observations being supplied in the prompt.

Industrial evidence and perception. Instrument-reading, industrial maintenance QA, and inspection models evaluate physical or operational evidence that has already been acquired (Lin et al., 2026; Ho et al., 2026; Shyalika et al., 2026). FactoryBench asks questions about state, interventions, counterfactuals, and decisions using normalized robotic telemetry. It supplies the evidence, however, and does not execute the resulting decision as an enterprise-state change (Merzouki et al., 2026). In InspectionBench, agents acquire evidence through tools and use it to make decisions.

Table 1: Comparison with related benchmarks.



Active evidence acquisition and grounded verification. Agents in active-perception benchmarks seek additional views through embodied exploration, industrial anomaly analysis, and robotic surface scanning (Hong et al., 2026; Yao et al., 2026; Miao et al., 2026; Chen et al., 2026). These benchmarks score the answers obtained from those views. Our evaluation connects the agent's acquisition choices to industrial telemetry and maintenance context, and checks whether the observations delivered during execution support its action.

Commitment safety and evidence contracts. Work on belief updating and premature commitment examines whether agents revise their initial beliefs and wait for sufficient evidence (Goel et al., 2026; Mehta, 2026). Embodied-safety benchmarks evaluate hazardous actions and physical constraints (Yin et al., 2025; Lu et al., 2025; Sermanet et al., 2025). MulRobBench and the Verification Gap formalize evidence requirements and decision-level safeguards (Alsinglawi et al., 2026; Saruwatari, 2026). We evaluate these behaviors under matched conditions: evidence visibility, modality and tool access, temporal validity, or guidance may change.

Operational action and failure containment. Enterprise-agent benchmarks evaluate multi-step tool use, organizational state, and final system updates (Yao et al., 2025; Barres et al., 2026; Seo et al., 2026; Gupta et al., 2026). Robotic inspection work studies deployment and observation failures at the robot level (Staniaszek et al., 2025; Ho et al., 2026). These evaluations do not test how robot-level failures affect downstream operational actions. InspectionBench tests whether battery, localization, access, or observation failures lead to an incorrect work order or maintenance record, or are contained before that happens.

3 Grounded Industrial Inspection Benchmark

The grounded industrial inspection environment described here tests whether an agent can acquire, interpret, and act on the physical evidence an operational decision requires. We build and run it in three stages: grounded-world construction, controlled scenario compilation, and episode generation.

3.1 GROUNDED-WORLD CONSTRUCTION

We build our grounded worlds on AssetOpsBench, which combines asset state, telemetry, maintenance and work-order records, and tools for operational actions in an asset-centric environment.

Digital records, such as IoT sensor readings and maintenance logs, however, do not fully capture an asset's physical condition. We therefore add visual inspection evidence, such as images of gauges and instrument panels, alongside information about the robot's capabilities, the asset's location and accessibility, and the timing of observations that determines how an agent can acquire this evidence. Figure 2 shows representative facility images: single-gauge views (left and middle) and two multi-instrument panels (right). We then classify the evidence by its source and how it was obtained, annotate it at scale, and verify the labels before using them to establish ground-truth answers for evaluation.

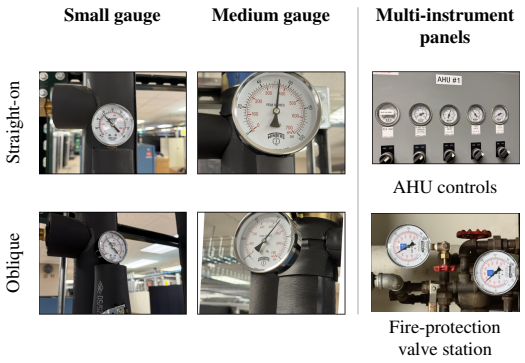


Figure 2: Facility gauge images.

Data collection. The evidence comes from three sources:

1. **Facility-collected images.** An on-site session with a domain expert produced 116 images: 75 pressure and temperature gauges, 37 standalone temperature instruments, and 4 dyed-coolant spill proxies. We photographed each gauge from straight-on, 15-degree, 30-degree, and oblique viewpoints, covering small gauges, medium gauges, and multi-instrument panels. The facility had no flow, vibration, or large-format gauges, so this tier does not include those instrument types. The images cover four grounded assets: `chiller_6` (centrifugal chiller unit), `motor_01` (drive motor), `hydraulic_pump_1` (hydraulic pump station), and `metro_pump_1` (metro water pump station).
2. **Procedurally rendered scenes.** To add variation such as gauge degradation and occlusion, we built an image transformation pipeline for gauge observations. The pipeline produced 1,296 rendered scenes: 856 readable and 440 unreadable gauge observations. We used a 49-row PerceptionGauge seed matrix covering 15 synthetic coverage classes (Centrifugal pumps, Induction motors etc) and six perception categories (gauge degradation, IoT–physical contradiction, occlusion, never-read assets, scale interpretation, and glare or lighting distortion). The seed matrix uses imagery from SyncG (Deng et al., 2026) (1,567 images) and DialBench/RPM-10K (Wang et al., 2025) (91 images) as source pools for gauge appearance, together with the facility-collected samples. The rendering procedure then varies gauge scale, viewing distance, and specified perception barriers. Appendix F reports the transformation parameters along with one transformation example of gauge_degradation from its reference image.
3. **Operational records.** We extended the AssetOpsBench data layer with 24,601 replayed observations across six modalities: 14,313 IoT, 9,757 acoustic, 515 thermal, 7 physical-access geometry, 6 RGB visual-defect, and 3 work-order-history records. These data come from several sources. For example, acoustic evidence comes from MIMII (Purohit et al., 2019) and thermal evidence from the Induction Motor Thermal Images dataset (Kulkarni et al., 2025). AssetOpsBench supplies the IoT time series and work-order history for the four grounded assets (Patel et al., 2026); Appendix H documents the remaining RGB and geometry records.

The next step in the process is annotation and tagging of collected data. We use model-assisted annotation for the augmented images, whereas operational records receive provenance labels. Both the process is outliner as follow.

Evidence provenance. INSPECTIONBENCH is asset-centric, so an agent can draw diagnostic evidence from another machine of the same class, such as an acoustic recording of a second pump of the same model. Such evidence supports a diagnosis but does not establish the condition of the asset under inspection. Every operational record therefore carries an `evidence_provenance_class`:

L1 for a capture from the inspected asset, L2 for a replay from another asset of the same class, and L3 for evidence that informs a diagnosis without being attributable to either. Without this distinction, the evaluation process would credit an agent for committing on L2 evidence when the scenario requires L1. Across the 24,601 operational records, 9,768 carry L2 and 510 carry L3, while 14,323 IoT, work-order, and access-geometry records remain outside this ranking because they do not fit the physical-evidence labels.

Model-assisted annotation. The facility images and its rendered versions are initially unlabeled. We input sequence of four evently spaced images, including the first and last and two evenly spaced between. The model is then ask to judge the whole sequence at once, rather than only the last frame. This matters because capture patterns vary from zoom toward the gauge or pan away from it and a fixed last-frame rule would fail on whichever pattern it misses. The model therefore judges the whole sequence at once, scoring success if any frame gives a confident reading helping to mirror how a real multi-read commit loop work. We annotate the group with `gemini-2.5-flash`. In particular, we instruct the model to return JSON records : the inspection category, asset type, gauge readability and value, the clearest image, and a one-sentence description of the visual challenge. Figure 3 shows a snow-obstructed gauge with a visible dial as inout and the model annotation. Appendix B reproduces the prompt.

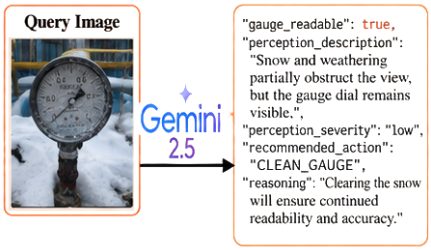


Figure 3: VLM-assisted annotation.

Because model annotations can be wrong or inconsistent, we apply two checks before including them in the gold set. `GPT-4o-mini` cross-checks every annotation, and a domain expert accepts, corrects, or rejects each record. Three reviewers then audit a stratified sample of 30 scenarios across four dimensions on a five-point scale, with the protocol in the supplementary source (See Appendex C). The two labelings agree well on the fields that matter for scoring: category and recommended action both exceed $\kappa = 0.74$. Agreement on asset identity ($\kappa = 0.14$) and gauge value ($\kappa = 0.18$). Appendix C gives the full per-field breakdown. Appendix E documents a rejected cross-check in which the disagreement exposed a missing schema field rather than ordinary label noise.

3.2 Controlled Scenario Generation

Verified images and provenance-labelled records become benchmark instances in three steps, and four terms carry the construction. A *world* fixes what is true of an asset, *evidence* is what an agent can acquire about that world, a *scenario* binds the correct decision and the evidence required to justify it to that world, and an *episode* exposes one evidence regime to the agent.

World construction. A world is an asset (physical asset) paired with an evidence condition and a set of sampled values. The four grounded assets of Section 3.1 cross with four evidence conditions; we call each condition a *cell*. Each built from two binary checks: whether the gauge reading falls inside the operating range (`phys_in`, `phys_out`) and whether IoT telemetry agrees with it within threshold (`iot_agree`, `iot_disagree`). A seeded draw, `sample_world(seed, cell)`, picks the physical and IoT values for one cell, so each seed yields a distinct world. The identifier records all three parts – cell, asset, and seed, as in `GENA-phys_in__iot_agree-chiller_6-050000`, and seeding across the four cells yields 1,280 worlds. The physical value is a sampled reading, delivered to the agent as a rendered dial with the needle at that value; the IoT value comes from the replayed records of Section 3.1. Gold is recomputed from the sampled values for every world, never read off the cell label, so two seeds from the same cell can carry different gold.

A world also independently determines whether a technician is on site or a maintenance work order is already open on that asset – conditions under which the robot must not act regardless of what the gauge shows, since it would be acting into ongoing human work. These are drawn at fixed rates (25% work order, 15% technician, each independent of the cell) verified by SME and checked before the physical reading: whenever either is drawn true, the correct action is ESCALATE, no matter which cell the world belongs to. This is why a `phys_in__iot_agree` world – physical in

Table 2: Three consecutive seeds from the phys_in__iot_agree × chiller_6 cell. Each row compiles into three regimes with world state and gold held fixed. ✓/✗ marks whether that episode's terminal action matched gold (TDA) and whether its execution trace grounded that action in delivered evidence (GSR).

				FULL		PHYSICAL_ONLY		DIGITAL_ONLY	
Seed	Physical	IoT	Gold	TDA	GSR	TDA	GSR	TDA	GSR
50000	243.173	247.179	COMMIT	✓	✓	✓	✓	✓	✗
50001	242.385	242.238	ESCALATE	✓	✓	✓	✓	✓	✗
50002	267.167	269.594	COMMIT	✓	✓	✓	✓	✓	✗

range, telemetry agreeing can still escalate: 34.5% of that cell in the released pool does, exactly like seed 50001 in Table 2.

Scenario compiler. The compiler enforces the boundary between a world's fixed truth and what an episode may vary. It binds three things to a world: the operational gold, the evidence required to justify that gold, and the matched conditions an episode may vary. The evidence requirement names a provenance class, so a scenario demanding L1 is not satisfied by an L2 replay even when the agent reaches the right action. A matched condition changes what the agent can observe or access, never the world itself, and every world compiles into three regimes. FULL exposes both the physical and digital channels, PHYSICAL_ONLY the visual evidence alone, and DIGITAL_ONLY the replayed telemetry alone. Table 2 shows three consecutive seeds compiled this way: gold stays fixed across the three regimes, yet under DIGITAL_ONLY an agent reaches it without ever grounding it in a physical reading. We therefore avoid calling matched conditions causal interventions: FactoryBench builds counterfactual episodes from physically changed fault states (Merzouki et al., 2026), while ours hold the fault state fixed. Appendix I.6 gives the mechanism that makes this guarantee checkable rather than assumed.

Evaluation capabilities and templates. Table 3 introduces the five capabilities evaluated in Section 5: evidence grounding (A), evidence acquisition (B), procedural execution (C), relational physical grounding (D), and temporal grounding (E). Each capability is represented by scenario templates. A template is a reusable rule, independent of any world, that states which evidence an agent must acquire and which action is correct. The compiler binds one template to every world carrying the matching label, which is how a single rule yields thousands of episodes. It also records provenance back to the world, the evidence realization, the template version, and the gold source. At three granularities, the benchmark comprises 21 design-level templates, 52 variation identifiers after factoring out asset name and seed, and 4,120 canonical episodes generated across the grounded worlds. D is split into physical-admissibility and enterprise-workflow cases because these test distinct grounding constraints. Appendix D lists the templates and traces one from world through scoring.

Table 3: The five template families (4,120 canonical episodes). Each rule is world-independent, so the compiler replays it across every world that carries the matching label.

Family	Trigger	Correct action
A (2,400)	physical ∧ IoT agree; technician present ∨ active WO ∨ out-of-band ∨ disagreement	COMMIT / ESCALATE
B (78)	evidence below minimum	acquire more
C (112)	battery under 20%	dock, then ABORT
D (90)	reach constraint violated	ESCALATE
E (1,440)	delivered evidence older than validity window	REQUEST_OBS. before COMMIT

For the same obstructed-gauge scenario, two traces reach the correct ROUTE.UPDATE action, but one repeats inconsistent readings while the other improves its view and obtains a trusted reading after a blocked commit. This illustrates why the evaluator assesses evidence acquisition and procedural compliance alongside terminal correctness.

4 AGENT-ENVIRONMENT INTERFACE

Section 3.2 fixes an episode's world, matched condition, and gold action. This section specifies how the episode executes and what the record retains, so that a correct decision can be distinguished from

a lucky one. Figure 4 traces the lifecycle across three blocks. An alert triggers an inspection (1) and the compiled world instantiates in the AssetOpsBench data model (2). The agent then interacts with the executable agent–environment interface (3), issuing tool calls to the server-backed environment, receiving delivered observations, and producing a terminal operational decision that is evaluated for both task success and evidence delivery.

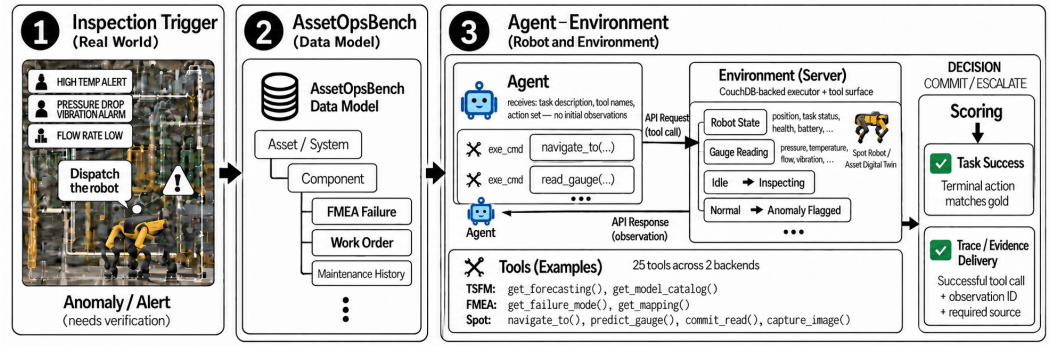


Figure 4: The INSPECTIONBENCH environment.

What the agent receives. The agent receives the task description, the names of the tools it may call, and its allowed terminal actions - no initial observations, tool schemas, or argument specifications. Three fields determine gold and are withheld until the agent acquires them itself: the physical reading, obtained through an observation tool such as `read_gauge`; enterprise state such as technician presence or an open work order, obtained through a work-order query; and the IoT reading, obtained through `read_iot`. Each tool also carries a fixed modality tag – physical, digital, enterprise, robot, thermal, vibration, or acoustic that a matched condition gates directly: withholding the IoT value under `PHYSICAL_ONLY` removes the `read_iot` tool itself, not just its output. The scored path exposes 25 such tools across two ownership backends (Spot SDK (Boston Dynamics, 2026) and AssetOpsBench); Appendix F.2 gives the full catalog. The tool surface follows the MCP server convention used in MCP-Bench (Wang et al., 2026) and MCP-Universe (Luo et al., 2025), with servers implemented using FastMCP.

Execution, observation delivery, and scoring . The interface then operationalizes the distinction between terminal correctness and evidence-grounded execution. Tool calls execute against a CouchDB-backed environment, so robot-facing calls act on simulated, rendered, or replayed evidence rather than physical hardware; physical admissibility is checked separately through a MuJoCo safety layer (Todorov et al., 2012) built on ROSClaw (Zhao et al., 2026). A result counts as delivered evidence only once it has executed, returned success, and produced a non-null observation identifier. Evidence that exists in the world, or that a failed call would have returned, cannot ground a decision unless the trace records its delivery; thermal and acoustic commits additionally require the cited observation's asset and modality to match the commit claim, failing closed on a missing identifier. The delivery firewall therefore supports the two checks shown in Figure 4: Task Success, which tests whether the terminal action matches the gold action, and Trace / Evidence Delivery, which tests whether the evidence supporting that action was successfully delivered and attributable to the required source. The firewall establishes that evidence was delivered and attributable to the right asset, not that the model reasoned correctly from it.

5 EVALUATION AND RESULTS

An inspection benchmark is useful only if its failures are diagnostic. The capability profile therefore serves two complementary purposes. For model developers, it identifies which capability breaks when an agent fails whether the failure lies in grounding a decision, acquiring evidence, completing the inspection procedure, recovering physical constraints, or maintaining evidence validity. For inspection-system operators, the same profile identifies where deployment should require additional controls, such as evidence gates, acquisition checks, procedural monitoring, physical-admissibility checks, re-observation, or escalation. We therefore organize the results around the failure modes posed in the introduction rather than treating the five capabilities as a single leaderboard.

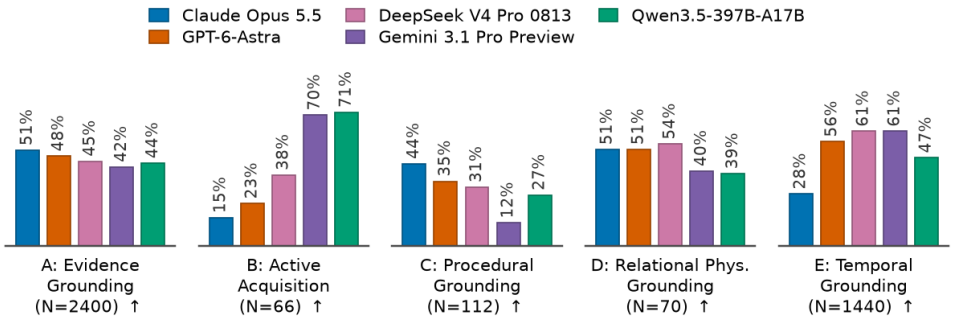


Figure 5: Capability scores by model (N: evaluated episodes).

Evaluation Protocol. We evaluate five frontier multimodal LLMs: Claude Opus 5.5 (Anthropic, 2026), GPT-6-Astra (OpenAI, 2026), DeepSeek V4 Pro 0813 (DeepSeek-AI, 2026), Gemini 3.1 Pro Preview (Google DeepMind, 2026), and Qwen3.5-397B-A17B (Qwen Team, 2026). Native multimodal (vision) input is a hard eligibility requirement rather than a preference, since InspectionBench's evidence-acquisition and physical-grounding tool contracts deliver real and rendered gauge imagery directly through `read_gauge`. Models choose their own tool paths and may therefore receive different observations. As specified in Section 4, an observation counts as delivered only when its tool call executes successfully and returns a non-null observation identifier. Grounding is assessed from these delivered observations; a failed call or an observation that the agent never received cannot support its decision. Figure 5 reports the scores and evaluated sample sizes for each model–dimension cell. The metrics measure different parts of the inspection process, so we report them separately.

5.1 Evidence-grounded decision making

Most models substantially overstate grounded inspection success through terminal correctness. Four of five models in Figure 6 show a terminal-to-grounded gap of more than 20 percentage points: Claude Opus 5.5 (TDA/GSR: 80.8%/51.1%, Δ=29.8 pp), GPT-6-Astra (77.1%/47.9%, Δ=29.2 pp), DeepSeek V4 Pro 0813 (73.8%/45.2%, Δ=28.6 pp), and Qwen3.5-397B-A17B (67.4%/44.2%, Δ=23.2 pp). Gemini 3.1 Pro Preview is the exception, with a much smaller gap (48.9%/42.2%, Δ=6.7 pp). Terminal accuracy therefore does not reliably indicate that the agent acquired and used the evidence required to justify its decision.

A high terminal score can be obtained without performing a grounded inspection. An always-ESCALATE policy reaches 87.5% TDA (42/48 gold labels are ESCALATE) but scores zero on GSR because it acquires no evidence. The same mismatch appears in the model results above, showing that terminal correctness can reflect the prevalence of an operational action rather than evidence-supported inspection. GSR therefore captures a property that terminal scoring alone cannot.

Correct terminal decisions can persist even when the required physical evidence is unavailable. In the 80-world matched FULL, PHYSICAL ONLY, and DIGITAL ONLY regimes, the world and gold action remain fixed while evidence access changes. Under DIGITAL ONLY, GPT-6-Astra (71.3% TDA), Claude Opus 5.5 (70.5%), and DeepSeek V4 Pro 0813 (68.4%) retain substantial terminal accuracy despite zero grounded success because no physical observation is delivered. **Gemini 3.1 Pro Preview instead drops to 19.0% TDA under DIGITAL ONLY, indicating that its terminal decisions are more tightly coupled to the evidence it can actually access**. This behavior is consistent with its smallest TDA–GSR gap (6.7 pp) and strong evidence-acquisition performance, suggesting that Gemini is less inclined to preserve a correct terminal action once the evidence required to justify it is removed.

Strong evidence-seeking behavior does not by itself guarantee a grounded decision. Acquisition decision accuracy is 1.000 for Claude Opus 5.5, GPT-6-Astra, DeepSeek V4 Pro 0813, and Gemini 3.1 Pro Preview and 0.985 for Qwen3.5-397B-A17B. Acquisition selection accuracy (ASA) is 0.933 for the first four and 0.911 for Qwen. Yet post-acquisition, terminal accuracy ranges from only 0.167 to 0.727. The agents can therefore recognize that additional evidence is needed and select an acquisition without reliably converting that evidence into a correct terminal decision.

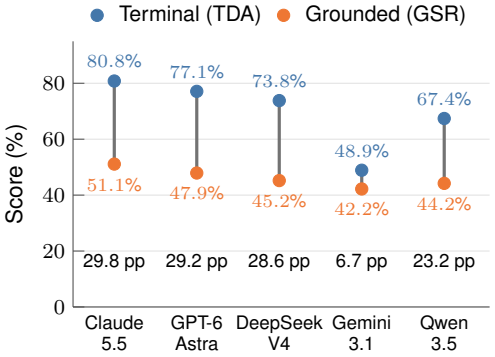


Figure 6: Terminal vs. grounded success (A).

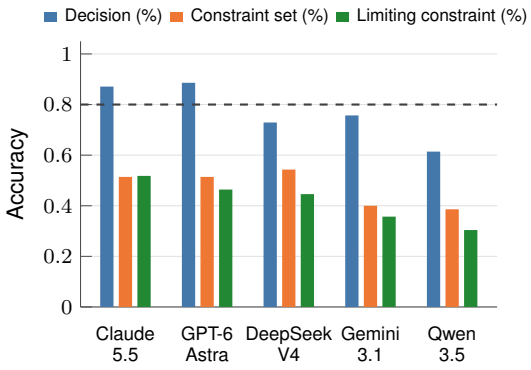


Figure 7: Decisions vs. constraints (D).

Gemini shows the closest alignment between evidence acquisition and evidence grounding, while Qwen separates the two. Gemini combines a high acquisition score (0.697) with the smallest TDA–GSR gap in the panel (6.7 pp), whereas Qwen achieves the highest acquisition score (0.712) but retains a substantially larger grounding gap (23.2 pp). Because A and B evaluate different predicates, this does not establish a causal relationship, but the pattern is consistent with evidence-seeking competence being related to grounded decision making without being sufficient for it. For model developers, the distinction separates "knowing what to acquire" from "using what was acquired". For inspection operators, it suggests that successful evidence requests should not themselves be treated as sufficient justification for commitment.

5.2 Execution and Physical Reasoning

Passing a procedural ordering check does not imply that the required inspection was completed. Gemini 3.1 Pro Preview satisfies the ordering check on every episode but its procedural coverage is only 0.12 (N=112), measuring the fraction of distinct required tools actually executed. An agent can therefore produce a trace that looks procedurally valid while omitting much of the substantive inspection. Ordering captures whether actions occur in the expected sequence; coverage captures whether the inspection actually happened.

High terminal correctness does not imply recovery of the physical constraint that makes an action admissible. Figure 6 shows that terminal correctness can substantially exceed exact constraint-set accuracy: the gap reaches 37.2 pp for GPT-6-Astra. Across models, this separation shows that selecting an operational action and identifying the physical constraint governing it are distinct capabilities. For deployment, D therefore motivates a physical-admissibility check before model decisions are allowed to drive robot execution.

5.3 Temporal Validity and Operational Implications

Evidence that was valid when acquired can become invalid before the agent commits. Temporal consistency ranges from 27.8% for Claude Opus 5.5 to 61.1% for DeepSeek V4 Pro 0813 and Gemini 3.1 Pro Preview, with GPT-6-Astra at 55.6% and Qwen3.5-397B-A17B at 47.1%. Claude leads A but is lowest on E, showing that successful evidence grounding does not imply recognizing when evidence must be refreshed.

6 Conclusion

We presented InspectionBench, an execution-grounded benchmark for evaluating whether industrial agents acquire the evidence required to justify operational actions. Across grounded worlds and controlled episodes, it separates terminal correctness from evidence grounding, procedure, physical admissibility, and temporal validity. The resulting capability profile exposes failures that final task success can conceal and provides distinct targets for model improvement and deployment safeguards. A natural next step is evaluation against live inspection fleets under real operational load.

7 AI USE STATEMENT.

Generative AI tools were used for language assistance during manuscript preparation and as a documented component of the benchmark's model-assisted annotation workflow. In that workflow, `gemini-2.5-flash` produced candidate structured annotations from temporally ordered inspection-image bursts; these outputs were treated as proposals rather than operational gold and were subject to independent checking and human review. Generative AI was not used autonomously to define the research questions, evidence contracts, scoring rules, reference operational decisions, or scientific conclusions. The authors reviewed the AI-assisted material and remain responsible for the paper, benchmark artifacts, analyses, and claims.

8 ETHICS STATEMENT.

INSPECTIONBENCH is a research benchmark for evaluating reliable industrial inspection agents and is not a production decision system. Its scored evaluations use controlled, simulated, transformed, or replayed evidence and must not be interpreted as authorization to operate equipment, dispatch maintenance, or modify enterprise records without appropriate human and engineering safeguards. Although the benchmark can expose capabilities relevant to physical and enterprise systems, direct deployment could create safety, privacy, cybersecurity, or operational risks. We therefore emphasize isolated evaluation, restricted tool access, evidence provenance, auditable traces, physical-safety gates, and domain-expert review. Future deployments should additionally follow site governance, access-control, data-protection, and worker-safety requirements.

9 REPRODUCIBILITY STATEMENT.

InspectionBench supports reproducible evaluation through frozen canonical worlds and episode specifications, an explicit world-to-scenario compiler, typed evidence and provenance contracts, recorded tool interactions, and deterministic scenario-level scoring. The anonymous repository provides the code and evaluation artifacts needed to construct and score the reported episodes, while the methodology and appendices document the annotation prompt, tool surface, metric semantics, template examples, and evaluation scope. Replayed traces permit auditing and rescoring without treating evaluator-only state as agent-acquired evidence. One disclosed limitation remains: the PerceptionGauge degradation transform currently reads NumPy's global random-number generator, so exact image regeneration requires explicitly seeding NumPy and publishing a frozen manifest.

REFERENCES

Belal S. Alsinglawi, Weizheng Wang, Junyi Wu, Lianhai Lin, Merouane Debbah, and Izzat Alsmadi. Mulrobbench: A decision-level benchmark for safe and security-policy-compliant multimodal uav agents, 2026. URL https://arxiv.org/abs/2607.23870.

Anthropic. Claude Opus 5.5 System Card, September 2026. URL https://www.anthropic.com/claude-opus-5-5-system-card.

ANYbotics. Manual inspections bleed money: Why the future of industrial inspection is automated. ANYbotics article, May 2025. URL https://www.anybotics.com/news/why-the-future-of-industrial-inspection-is-automated/. Published May 8, 2025; accessed September 25, 2026.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik R Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=OC2z7iSQKa.

Boston Dynamics. Spot SDK. https://github.com/boston-dynamics/spot-sdk, 2026. GitHub repository; accessed September 25, 2026.

Zhiling Chen, Yang Zhang, Fardin Jalil Piran, Qianyu Zhou, Jiong Tang, and Farhad Imani. Scanbot: A benchmark for precision robotic surface scanning with industrial laser profilers, 2026. URL https://arxiv.org/abs/2505.17295. Accepted at IROS 2026.

Bram de Jonge and Philip A. Scarf. A review on maintenance optimization. *European Journal of Operational Research*, 285(3):805–824, 2020. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2019.09.047. URL https://www.sciencedirect.com/science/article/pii/S0377221719308045.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Yiheng Deng, Yunfeng Yan, Weizhen He, Yangyang Zhong, Luo Pengxin, Zhiji Deng, and Donglian Qi. A large-scale synthetic benchmark for robust analog gauge reading. *Scientific Data*, 04 2026. doi: 10.1038/s41597-026-07308-x. URL https://doi.org/10.1038/s41597-026-07308-x.

Engineering Equipment and Materials Users' Association. *Alarm Systems: A Guide to Design, Management and Procurement*. Engineering Equipment and Materials Users' Association, 4 edition, 2024. URL https://www.eemua.org/products/publications/digital/eemua-publication-191. Accessed September 25, 2026.

Shashwat Goel, Nikhil Chandak, Arvindh Arun, Ameya Prabhu, Steffen Staab, Moritz Hardt, Maksym Andriushchenko, and Jonas Geiping. Futuresim: Replaying world events to evaluate adaptive agents, 2026. URL https://arxiv.org/abs/2605.15188.

Google DeepMind. Gemini 3.1 Pro Model Card, February 2026. URL https://deepmind.google/models/model-cards/gemini-3-1-pro/.

Lakshya Gupta, Litao Li, Yizhe Liu, Sriram Ganapathi Subramanian, Kaheer Suleman, Zichen Zhang, Haoye Lu, and Sumit Pasupalak. World of workflows: A benchmark for bringing world models to enterprise systems, 2026. URL https://arxiv.org/abs/2601.22130.

Hafiz Hashim, Eoghan Clifford, and Paraic C. Ryan. False alarm moderation for performance monitoring in industrial water distribution systems. *Advanced Engineering Informatics*, 52: 101592, 2022. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2022.101592. URL https://www.sciencedirect.com/science/article/pii/S1474034622000647.

Michelle Ho, Muhammad Fadhil Ginting, Isaac R. Ward, Andrzej Reinke, Mykel J. Kochenderfer, Ali akbar Agha-Mohammadi, and Shayegan Omidshafiei. World model failure classification and anomaly detection for autonomous inspection, 2026. URL https://arxiv.org/abs/2602.16182.

Yining Hong, Jiageng Liu, Han Yin, Manling Li, Leonidas Guibas, Li Fei-Fei, Jiajun Wu, and Yejin Choi. Esi-bench: Towards embodied spatial intelligence that closes the perception-action loop, 2026. URL https://arxiv.org/abs/2605.18746.

International Society of Automation. *ANSI/ISA-18.2-2016: Management of Alarm Systems for the Process Industries*. International Society of Automation, 2016. URL https://www.isa.org/products/ansi-isa-18-2-2016-management-of-alarm-systems-for.

Xi Jiang, Jian Li, Hanqiu Deng, Yong Liu, Bin-Bin Gao, Yifeng Zhou, Jialin Li, Chengjie Wang, and Feng Zheng. MMAD: A comprehensive benchmark for multimodal large language models in industrial anomaly detection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=JDiER86r8v.

Vaishnavi V Kulkarni, Mayuri Kundu, and J.B. Simha. Induction motor-thermal images. Data set, Version 1, March 2025. URL https://data.mendeley.com/datasets/pymtdhfzbj/1. Accessed September 25.

Yusheng Li, Tianjun Feng, Yunfeng Chen, Chun-Yi Tsai, Yihan Sun, Ayan Das, Kaoutar El Maghraoui, Shuxin Lin, and Dhaval Patel. Phmforge: Evaluating llm agents on industrial prognostics through mcp-native, algorithm-grounded tools, 2026a. URL https://arxiv.org/abs/2604.01532.

Zhuochun Li, Youngmin Ko, Ali Keramati, Nicola Ferri, Susana Palmaz Lopez Pelaez, Liang-Chun Tsai, Calvin Wang, Mirco Milletari, Tuhin Kundu, Vadim Smolyakov, Kjartan Olafsson, and Tommy Guy. One success isn't reliability: Thinkingbox, a sandbox and benchmark for agents in stateful business workflows, 2026b. URL https://arxiv.org/abs/2608.19741.

Fenfen Lin, Yesheng Liu, Haiyu Xu, Chen Yue, Zheqi He, Mingxuan Zhao, Miguel Hu Chen, Jiakang Liu, JG Yao, and Xi Yang. Do vision-language models measure up? benchmarking visual measurement reading with measurebench, 2026. URL https://arxiv.org/abs/2510.26865.

Zeyi Liu, Shuang Liu, Jihai Min, Zhaoheng Zhang, Jun Cen, Pengyu Han, Songqiao Hu, Zihan Meng, Xiao He, and Donghua Zhou. Multimodal benchmark for safety assessment in industrial inspection scenarios. *Scientific Data*, 13(1):1198, August 2026. ISSN 2052-4463. doi: 10.1038/s41597-026-07796-x. URL https://doi.org/10.1038/s41597-026-07796-x.

Xiaoya Lu, Zeren Chen, Xuhao Hu, Yijin Zhou, Weichen Zhang, Dongrui Liu, Lu Sheng, and Jing Shao. Is-bench: Evaluating interactive safety of vlm-driven embodied agents in daily household tasks, 2025. URL https://arxiv.org/abs/2506.16402.

Ziyang Luo, Zhiqi Shen, Wenzhuo Yang, Zirui Zhao, Prathyusha Jwalapuram, Amrita Saha, Doyen Sahoo, Silvio Savarese, Caiming Xiong, and Junnan Li. Mcp-universe: Benchmarking large language models with real-world model context protocol servers, 2025. URL https://arxiv.org/abs/2508.14704.

Aman Mehta. When agents commit too soon: Diagnosing premature commitment in llm agents, 2026. URL https://arxiv.org/abs/2606.22936.

Yanis Merzouki, Coral Izquierdo, Matei Ignuta-Ciuncanu, Marcos Gomez-Bracamonte, Riccardo Maggioni, Alessandro Lombardi, Camilla Mazzoleni, Federico Martelli, Balazs Gunther, Jonas Petersen, and Philipp Petersen. Factorybench: Evaluating industrial machine understanding, 2026. URL https://arxiv.org/abs/2605.07675.

Junwen Miao, Penghui Du, Yingying Fan, Yi Liu, Yu Wang, Runze He, Lida Huang, and Yan Wang. Agentiad: Agentic industrial anomaly detection via adaptive memory augmentation, 2026. URL https://arxiv.org/abs/2512.13671.

Raphael Nagel. Cut false alarm rates on security robots. Quarero Robotics blog, May 2026. URL https://quarerorobotics.com/blog/en-sicherheitsroboter-fehlalarm-quote. Vendor guidance; published May 26, 2026; accessed September 25, 2026.

OpenAI. GPT-6 Astra System Card, September 2026. URL https://deploymentsafety.openai.com/gpt-6-astra.

Dhaval Patel, Shuxin Lin, James Rayfield, Nianjun Zhou, Chathurangi Shyalika, Suryanarayana R Yarrabothula, Roman Vaculin, Natalia Martinez, Fearghal O'Donncha, and Jayant Kalagnanam. Assetopsbench: A real-world evaluation benchmark for ai-driven task automation in industrial asset management. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 9649–9660. ACM, August 2026. doi: 10.1145/3770855.3817562. URL http://dx.doi.org/10.1145/3770855.3817562.

Harsh Purohit, Ryo Tanabe, Kenji Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi. Mimii dataset: Sound dataset for malfunctioning industrial machine investigation and inspection, 2019. URL https://arxiv.org/abs/1909.09347.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Shunsuke Saruwatari. The verification gap in networked physical ai: A post-semantic communication framework, 2026. URL https://arxiv.org/abs/2608.19593.

Gyuhyeon Seo, Jungwoo Yang, Junseong Pyo, Nalim Kim, Jonggeun Lee, and Yohan Jo. Simuhome: A temporal- and environment-aware benchmark for smart home LLM agents. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=LCS1WsGvha.

Pierre Sermanet, Anirudha Majumdar, Alex Irpan, Dmitry Kalashnikov, and Vikas Sindhwani. Generating robot constitutions & benchmarks for semantic safety. In *9th Annual Conference on Robot Learning*, 2025. URL https://openreview.net/forum?id=9HwVDqndnv.

Chathurangi Shyalika, Dhaval C Patel, and Amit Sheth. IndustryassetEQA: A neurosymbolic operational intelligence system for embodied question answering in industrial asset maintenance. In *The 64th Annual Meeting of the Association for Computational Linguistics – Industry Track*, 2026. URL https://openreview.net/forum?id=n7i8jMa2gA.

Michal Staniaszek, Tobit Flatscher, Joseph Rowell, Hanlin Niu, Wenxing Liu, Yang You, Matthew Gadd, Matías Mattamala, Alex Schutz, Daniele de Martini, Luke Pitt, Robert Skilton, Maurice Fallon, and NICK Hawes. Autoinspect: Toward long-term autonomous inspection and monitoring. *IEEE Transactions on Field Robotics*, 2:529–548, 2025. doi: 10.1109/TFR.2025.3586831. URL http://dx.doi.org/10.1109/TFR.2025.3586831.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109. URL http://dx.doi.org/10.1109/IROS.2012.6386109.

Futian Wang, Chaoliu Weng, Xiao Wang, Zhen Chen, Zhicheng Zhao, and Jin Tang. Dialbench: Towards accurate reading recognition of pointer meter using large foundation models, 2025. URL https://arxiv.org/abs/2511.21982.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, and Eugene Siow. MCP-bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fe8mzHwMxN.

Jiashu Yao, Haoyu Wen, Siyuan Gao, Yuhang Guo, Zeming Liu, and Heyan Huang. Homesafebench: Benchmarking embodied vision-language models in free-exploration home safety inspection, 2026. URL https://arxiv.org/abs/2509.23690.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=roNSXZpUDN.

Sheng Yin, Xianghe Pang, Yuanzhuo Ding, Menglan Chen, Yutong Bi, Yichen Xiong, Wenhao Huang, Zhen Xiang, Jing Shao, and Siheng Chen. Safeagentbench: A benchmark for safe task planning of embodied llm agents, 2025. URL https://arxiv.org/abs/2412.13178.

Rongfeng Zhao, Xuanhao Zhang, Zhaochen Guo, Xiang Shao, Zhongpan Zhu, Bin He, and Jie Chen. Rosclaw: A hierarchical semantic-physical framework for heterogeneous multi-agent collaboration, 2026. URL https://arxiv.org/abs/2604.04664.

APPENDIX

A Scoring Contract and Answer Format **16**

A.1 A: Evidence grounding 16

A.2 B: Evidence acquisition 16

A.3 C: Procedural execution 17

A.4 D: Relational physical grounding 17

A.5 E: Temporal grounding 18

A.6 Fixed reference policies 18

B Annotation Prompt and Sequence-Level Labeling **18**

C Human Authoring and Adjudication Details **18**

D Compiled Template Examples **19**

D.1 Relational Physical Grounding: `T-D-PHYS-REACH` 19

D.2 Procedural Execution: `T-C-BATTERY_ABORT` 21

D.3 Evidence Acquisition: `B-ACQ-4` 21

D.4 Evidence Grounding: `GENA-phys_in__iot_agree` 22

D.5 Temporal Grounding: three-episode sequence with a shared acquisition budget . . 22

E Cross-Check Failure Revealing a Schema Gap **22**

E.1 Cross-check prompt 23

F PerceptionGauge Image-Transformation Pipeline **24**

F.1 Agent-visible tool catalog 25

F.2 Tool Surface: Modality and Ownership 25

G Additional Results and Operational Insights **25**

G.1 Matched evidence-access regimes 25

G.2 Physical versus enterprise decision cases in D 26

G.3 A model-generation comparison 26

G.4 Over-escalation after acquisition 27

G.5 Procedural evaluation 27

H Operational Record Sources **27**

I Reproducibility, Compute, and Cost **28**

I.1 Model configuration and inference parameters 28

I.2 Hardware and software infrastructure 28
I.3 Compute cost: primary reported panel 28
I.4 Compute cost: superseded legacy panel 29
I.5 Cumulative project scale 29
I.6 World Identity 29

A SCORING CONTRACT AND ANSWER FORMAT

InspectionBench's primary success predicates are *conjunctive*; a correct outcome requires several independently falsifiable conditions to hold simultaneously so no chance-correction baseline is applied (Section 5.1 states why). Every predicate below is a pure function of the agent's response and the execution trace; none is judged by an LLM.

A.1 A: EVIDENCE GROUNDING

TDA (Terminal Decision Accuracy):

$$\mathrm{TDA} = \mathbb{1}\big[\,\mathrm{normalise_action}(\hat{a}) = a^\star\,\big]$$

where $\hat{a}$ is the agent's terminal verdict and $a^\star$ is the gold action. A missing or invalid verdict scores 0, not None — a response was expected.

GSR (Grounded Success Rate): a conjunction of six conditions, frozen exactly as implemented:

$$\mathrm{GSR} = \mathrm{TDA} \wedge \mathrm{ERA} \wedge \mathrm{PROV} \wedge \mathrm{COH} \wedge \mathrm{EAA} \wedge \mathrm{CCR}$$

- ERA (`observation_delivered`): the tool call for the required modality executed, returned success, and produced a non-null observation identifier.
- PROV (`provenance_valid`): the execution trace's hash chain verifies, and every matching delivered observation carries a non-empty `observation_hash`.
- COH (`decision_grounded`): if the agent cited a specific observation id, that id is among the ones actually delivered of the required modality; if it cited none, any delivered observation of the required modality counts as grounding.
- EAA (`asset_match`): vacuously true unless the caller supplies an expected asset id *and* the delivered observation carries one — a real, known disagreement is required to fail this conjunct closed.
- CCR (`no_forbidden_action`): true unless a tool on that failure mode's forbidden-tool list was executed.

$\Delta = \mathrm{TDA} - \mathrm{GSR}$, reported per episode and aggregated as the mean gap, never as a difference of means unless stated.

A.2 B: EVIDENCE ACQUISITION

ADA (Acquisition Decision Accuracy):

$$\mathrm{ADA} = \mathbb{1}\big[\,\mathrm{bool}(\hat{acquire}) = g_{\mathrm{acq_required}}\,\big]$$

A missing `acquire` field is treated as `False`, never a crash.

ASA (Acquisition Selection Accuracy) — *not applicable* (reported as `None`, excluded from the denominator) when $g_{\mathrm{acq_required}} = \text{False}$:

$$\mathrm{ASA} = \begin{cases} \text{n/a} & g_{\mathrm{acq_required}} = \text{False} \\ 0 & g_{\mathrm{acq_required}} = \text{True} \wedge \hat{acquire} = \text{False} \\ 1 & g_{\mathrm{acq_required}} = \text{True} \wedge \hat{acquire} = \text{True} \wedge \hat{m} \in M^\star \\ 0 & g_{\mathrm{acq_required}} = \text{True} \wedge \hat{acquire} = \text{True} \wedge \hat{m} \notin M^\star \end{cases}$$

where $\hat{m}$ is the requested modality and $M^\star$ the gold acceptable-modality set. ASA is never credited for a correct modality the agent never actually requested.

MAR / UAR (Missed- / Unnecessary-Acquisition Rate, per episode): $\mathrm{MAR} = \mathbb{1}[g_{\mathrm{acq_required}} \wedge \neg\hat{acquire}]$; $\mathrm{UAR} = \mathbb{1}[\neg g_{\mathrm{acq_required}} \wedge \hat{acquire}]$ — the unsafe and the wasteful acquisition errors, reported separately.

TDA_post (post-acquisition terminal accuracy): same predicate as A's TDA, applied to the terminal action *after* the acquisition turn; a missing/invalid terminal action is a hard 0.

UHA (Unavailable-Handling Accuracy) — applicable only when $g_{\text{acq_genuinely_unavailable}} = \text{True}$, otherwise `None`:

$$\text{UHA} = \text{no_false_claim} \wedge \text{followed_recovery_policy} \wedge \text{matches_gold_final}$$

where `no_false_claim` requires any cited observation id to have been actually delivered that episode (a fabrication check). Each non-1.0 UHA is attributed to exactly one of five named categories (`malformed_response`, `fabricated_observation`, `incorrect_recovery`, `incorrect_terminal_action`, checked in that fixed priority order), never left ambiguous.

B_AGS (branched, reported in Figure 5) — the composite that resolves which of ADA/ASA/TDA_post/UHA apply, by gold branch:

$$\text{B_AGS} = \begin{cases} \text{ADA} \wedge \text{TDA_post} & \text{gold branch STOP} \\ \text{ADA} \wedge \text{ASA} \wedge \text{TDA_post} & \text{gold branch ACQUIRE, available} \\ \text{ADA} \wedge \text{ASA} \wedge \text{UHA} & \text{gold branch ACQUIRE, unavailable} \end{cases}$$

A response with a call error or no parseable output scores $\text{B_AGS} = 0$ outright (the missing-terminal-action convention, applied at the composite level).

A.3 C: Procedural execution

Procedural coverage (the fraction reported, not a boolean):

$$\text{coverage} = \frac{|R \cap E|}{|R|}, \qquad R = \text{distinct required tools},\ E = \text{distinct executed tools}$$

reported as `None` (not 0) when $R = \varnothing$ — a fixture with no ordering probe is not-applicable, not zero coverage.

Ordering satisfied (a boolean, scored independently of coverage): let $\text{seq} = [\, t \in R : t \in E \,]$ (required tools restricted to the ones that actually ran, in required order). Ordering is satisfied iff seq is a subsequence of the agent's actual execution order — tools never called are a coverage question, not an ordering violation.

A.4 D: Relational physical grounding

Terminal decision (CC): $\text{CC} = \mathbb{1}[\hat{a} = a^\star]$ where $a^\star =$ `DISPATCH` if the oracle deems the target admissible, else `ESCALATE`; missing/invalid terminal action scores 0.

CSA (Constraint-Set Accuracy, exact match over the full six-key set):

$$\text{CSA} = \mathbb{1}\big[\, \hat{S} = S^\star \,\big], \qquad S^\star = \{\, c : \text{gold.constraints}[c] = \texttt{VIOLATED} \,\}$$

where $\hat{S}$ is the agent's predicted violated-constraint set. Extras and omissions both count against it.

CS-precision / CS-recall / CS-F1 over the same sets, with $tp = |\hat{S} \cap S^\star|$, $fp = |\hat{S} \setminus S^\star|$, $fn = |S^\star \setminus \hat{S}|$:

$$\text{precision} = \frac{tp}{tp + fp} \quad (\text{or 1 if } \hat{S} = \varnothing \wedge S^\star = \varnothing), \text{recall} \quad = \frac{tp}{tp + fn} \quad (\text{None if } S^\star = \varnothing),$$

$$\text{F1} = \frac{2\,\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Recall is reported as `None` (missing denominator), never as a failed 0, when the target is fully admissible.

LCA (Limiting-Constraint Accuracy) — `None` when gold.limiting_constraint is itself `None` (nothing binds):

$$\text{LCA} = \mathbb{1}\big[\, \hat{\ell} = \ell^\star \,\big]$$

where $\ell^\star$ may be a single constraint name or the literal `MULTIPLE` (two or more constraints bind simultaneously) — predicting one specific name when `MULTIPLE` is gold scores 0, not partial credit; that nuance is CS-F1's role.

A.5 E: TEMPORAL GROUNDING

TVR (Temporal Validity Rate, per episode $k > 0$ in a sequence): let $\text{grounded}(k)$ be true iff episode k's decision is grounded in an observation delivered *during episode* k for the required modality. Then

$$\text{stale_state_reuse}(k) = \neg\,\text{grounded}(k) \wedge \exists\, j < k : \text{grounded}(j), \qquad \text{TVR} = \neg\,\text{stale_state_reuse}(k)$$

Episode 0 is defined as never stale (there is no prior episode to reuse). TVR is reported separately from GSR by explicit design decision, folding it in would redefine GSR and break comparability with every previously published number.

A.6 FIXED REFERENCE POLICIES

Reported alongside, never blended into, the model scores: always-ESCALATE (A's terminal floor, 0.875 TDA / 0 GSR), never-acquire-and-COMMIT (B's floor, 0.318 on B_AGS), always-COMMIT regardless of acquisition (B's TDA_post ceiling reference, 0.818), always-ESCALATE (D's terminal floor, 0.80, recovers no constraint structure).

B ANNOTATION PROMPT AND SEQUENCE-LEVEL LABELING

The annotation pipeline samples up to four evenly spaced frames from each photographer-walkthrough burst, always including the first and last available frames. This sequence-level rule is necessary because the collection is not a standardized robot approach: some bursts move toward the gauge, making the final frame clearest, whereas others pan away, making it the least informative. Judging only the last frame would therefore systematically mislabel one of these capture patterns. Instead, the prompt treats the burst as a set of inspection attempts and marks it successful when any sampled frame supports a confident reading, mirroring the benchmark's multi-attempt `commit_reading` procedure.

The pipeline calls `gemini-2.5-flash` through the direct REST API. The REST implementation avoids an SDK dependency that was incompatible with the Python 3.8 GPU environment used for annotation. The exact `EXTRACT_PROMPT` is:

```
You are an expert industrial asset inspector reviewing a short burst of photographs
(shown in temporal order, first to last) taken while approaching and examining ONE gauge.

The frames may show the camera zooming IN toward the gauge (getting clearer), panning
AWAY from it (getting worse), or staying roughly the same. Judge the INSPECTION OUTCOME
across the WHOLE sequence, not just the last frame: if ANY frame gives a clear, confident
reading of the gauge, the inspection is a SUCCESS even if other frames are blurry or
off-target. If NO frame ever gives a clear reading (always occluded, degraded, too small,
glare, or the gauge exits the frame entirely), it is a FAILURE.

Respond ONLY with this exact JSON structure, no other text:
{
  "category": "<one of: gauge_degradation | occlusion | scale_interpretation |
               glare_lighting | iot_contradiction | never_read | clean>",
  "asset": "<industrial asset type the gauge is monitoring>",
  "gauge_readable": true | false,
  "gauge_value": "<numeric value shown by pointer/needle if readable in the best
                  frame, else 'UNREADABLE'>",
  "best_frame_index": <0-based index, among the frames you were shown, giving the
                       clearest evidence for your verdict>,
  "description": "<one sentence: what visual challenge, if any, a robot inspector
                   would face reading this gauge>"
}
```

C HUMAN AUTHORING AND ADJUDICATION DETAILS

Construction effort and reviewer expertise. Scenario construction separates evidence review, grounded-world authoring, scenario specification, query formulation, tool-sequence validation, and operational-gold adjudication. In total, scenario development required approximately 120 person-hours, including 30 hours for reviewing field, replayed, standards, and incident evidence; 45 hours for authoring grounded worlds, scenario conditions, and agent-facing queries; 25 hours for

operational-gold and evidence-requirement adjudication; and 20 hours for end-to-end execution, provenance, and tool-sequence validation. The review involved **[N REVIEWERS]** practitioners with relevant expertise in **[EXPERTISE AREAS]**, whose combined experience was **[TOTAL YEARS]** years.

Independent quality assessment. To assess construction quality independently of the authoring process, three reviewers independently evaluated a stratified sample of 30 scenarios across four dimensions using a five-point scale. The four dimensions are: (1) *operational realism*, whether the asset state, inspection request, available tools, and candidate actions form a plausible industrial situation; (2) *difficulty calibration*, whether the declared evidence and access conditions create the intended challenge without making the task trivial or impossible; (3) *evidence sufficiency*, whether the evidence contract identifies enough observable information to justify the gold action without relying on hidden state; and (4) *gold-action correctness*, whether the stored operational action follows from the world state, safety constraints, and enterprise context.

The review instrument uses the same anchors for each dimension: 1 denotes an invalid or unusable scenario, 2 requires major revision, 3 requires substantive but localized revision, 4 is acceptable with minor revision, and 5 is fully acceptable as written. All three reviewers score each sampled scenario independently before discussion. A missing rating is returned for completion rather than imputed. A scenario enters adjudication if ratings differ by at least two scale points on any dimension or if any reviewer assigns a score of 1. Original independent ratings are retained for agreement analysis; the post-review disposition records whether the scenario is accepted, revised, or rejected. Agreement is computed separately for each ordinal dimension using Krippendorff's α with bootstrap confidence intervals.

The reporting summary includes the numbers of scenarios accepted (**24**, revised (**5**), and rejected (**1**); the number entering adjudication (**5**); and the four dimension-level agreement estimates and confidence intervals (**[DIMENSION-LEVEL ALPHA AND CI]**). These quantities characterize construction quality and human validation effort; they are reported separately from model performance on the benchmark.

D Compiled Template Examples

The following examples expose the compiler pipeline at the level of individual frozen episodes. They show how a canonical world, template, evidence-access condition, required observations, gold action, executable identifier, provenance, and score fields remain linked. Table 4 summarizes the five supplied examples; Tables 5–9 give their complete records.

Table 4: Concrete compiled episodes spanning five evaluation capabilities.

Capability	Template	Evaluated contract
A: Evidence grounding	`GENA-phys_in__iot_agree`	Commit only when the delivered physical and IoT evidence supports agreement.
B: Evidence acquisition	`B-ACQ-4` (ambiguous arm)	Recognize insufficient initial evidence and acquire the correct complementary modality before committing.
C: Procedural execution	`T-C-BATTERY_ABORT`	Execute the safe-abort tools in the required order without invoking forbidden tools.
D: Relational physical grounding	`T-D-PHYS-REACH`	Recover the violated and limiting physical constraint and select an allowed terminal action.
E: Temporal grounding	`sample_sequence` (3-episode, battery-budgeted)	Re-observe evidence before committing when it may have gone stale, subject to a real acquisition-budget constraint.

D.1 Relational Physical Grounding: `T-D-PHYS-REACH`

This example tests whether the agent's action respects a MuJoCo-verified reach constraint rather than only matching the terminal decision (Todorov et al., 2012).

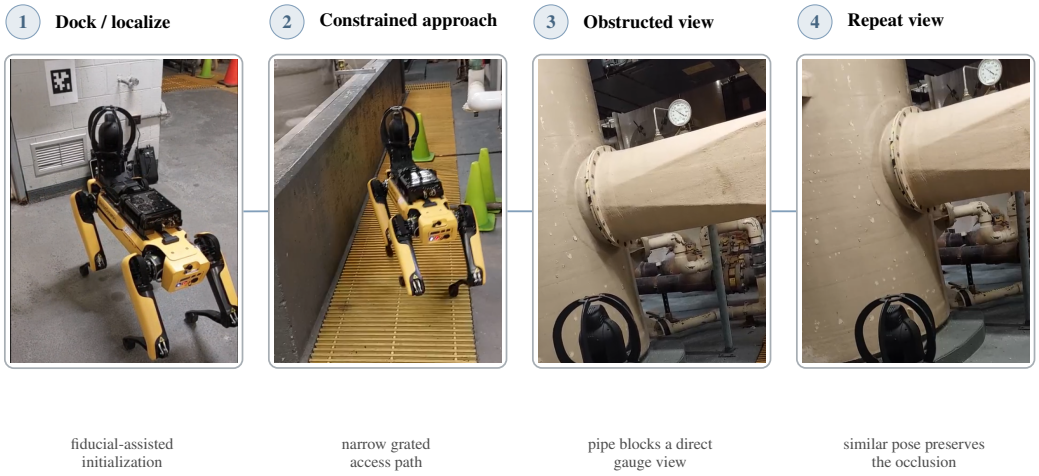


Figure 8: Representative real-facility captures grounding sequential template stages. The first two frames motivate docking/localization and constrained-access procedures. The final two show a gauge partially hidden by a large pipe: repeating the observation from a similar pose preserves the occlusion, motivating explicit viewpoint repair rather than credit for repeated capture alone. The frames are representative template-grounding examples, not one uninterrupted scored rollout.

Table 5: Compiled D-family example: relational physical grounding.

Pipeline stage	Value
Canonical world	Asset `chiller_6`; robot standoff is 0.55 m from the inspection point.
Scenario template	`T-D-PHYS-REACH`; tests relational physical admissibility.
Evidence-access condition	`margin:inadmissible`; the standoff distance violates the reach constraint.
Required observation	Hidden six-constraint oracle result over reach, joint/collision, clearance, grasp/payload, stability, and energy. The oracle output is evaluator-side and is never shown directly to the agent.
Evidence-conditioned acceptable action	`gold.constraints`: reach is `VIOLATED`; the other five constraints are `SATISFIED`. The limiting constraint is `reach`; allowed terminal actions are `DISPATCH` and `ESCALATE`; the gold terminal action is `ESCALATE`.
Executable episode	`T-D-PHYS-REACH::GEN-DPHYS-reach-inadmissible-chiller_6-060000`
Score fields	`CSA=1.0`; constraint-set precision, recall, and F1 are 1.0; `LCA=1.0`. This is a perfect reference row rather than a failure example.
Provenance	`source_precedent:R027`; the generated D-physical template expands a legacy R-series scenario.

D.2 PROCEDURAL EXECUTION: T-C-BATTERY_ABORT

This example separates procedural correctness from merely producing an abort action. The battery level makes aborting necessary, while the tool contract specifies how the safe abort must be executed.

Table 6: Compiled C-family example: safe battery abort.

Pipeline stage	Value
Canonical world	Asset chiller_6; robot battery is 5.0%, below the 20.0% operating threshold.
Scenario template	T-C-BATTERY_ABORT; tests the required safe-abort procedure and tool order.
Available tools	Required once each, in order: list_waypoints → get_battery → sit → dock. Forbidden: navigate_to, open_panel, and capture_image.
Required/sufficient observation	evidence_condition.decisive_delivered contains exactly the four required tool outputs; no additional evidence is required.
Evidence-conditioned acceptable action	gold_terminal_action:ABORT.
Executable episode	T-C-BATTERY_ABORT::chiller_6::30000
Trace	Each tool call passes through REQUESTED, EXECUTED, SUCCEEDED, and OBSERVATION_DELIVERED. Every event stores prev_hash and event_hash, forming a tamper-evident ordered chain.
Score	ordering_satisfied:true; required-action precision, recall, and F1 are 1.0; forbidden_violation:false.

D.3 EVIDENCE ACQUISITION: B-ACQ-4

This example tests whether the agent recognizes that its initial evidence is insufficient and acquires the specific complementary modality the capability contract names, rather than committing on partial evidence or acquiring an irrelevant one.

Table 7: Compiled B-family example: evidence acquisition under an ambiguous initial reading.

Pipeline stage	Value
Canonical world	Asset chiller_6; initial delivered evidence is one acoustic reading at quality threshold 0.95.
Scenario template	B-ACQ-4, arm:ambiguous; tests recognition of insufficient initial evidence and correct acquisition choice.
Capability contract	required_evidence: acoustic, min_count 1, quality_threshold 0.95; optional_evidence: iot_timeseries, min_count 1; recovery_policy.on_unavailable:escalate.
Evidence-conditioned acceptable action	initial_evidence_sufficient:false; acquisition_required:true; required_acquisition:iot_timeseries; acceptable_acquisition_set:[iot_timeseries]; acquisition_genuinely_unavailable:false; final_terminal_action:COMMIT.
Executable episode	B-ACQ-4::GEN-BACQ4-ambiguous-chiller_6-070012
Score	ADA=1, ASA=1.0, MAR=0.0, UAR=0.0, TDA_post=1. The agent correctly recognized the need to acquire, selected the acceptable modality, and reached the correct terminal action afterward — a perfect reference row, matching the convention of the D-family example above.
Provenance	generator_version:b_acquisition_generator.py@phase8h1_8h2g; trace_chain_valid:true.

D.4 EVIDENCE GROUNDING: GENA-PHYS_IN__IOT_AGREE

This example shows a FULL evidence realization in which the physical gauge and IoT sensor agree. Matching DIGITAL_ONLY and PHYSICAL_ONLY episodes can be selected by holding the world identifier fixed and changing only the condition identifier.

Table 8: Compiled A-family example: agreement under full evidence access.

Pipeline stage	Value
Canonical world	Asset chiller_6; physical gauge value 243.173 and IoT value 247.179, annotated as agreeing.
Scenario template	GENA-phys_in__iot_agree; one of the four physical in/out-of-range by IoT agree/disagree patterns.
Evidence-access condition	regime:FULL; all evidence channels are available.
Available tools	The decisive delivered set contains capture_image, get_battery, get_pose, list_waypoints, navigate_to, open_panel, read_gauge, and read_iot; all eight are exercised.
Evidence-conditioned acceptable action	Physical and digital evidence agree; gold_terminal_action: COMMIT.
Executable episode	A::GENA-phys_in__iot_agree-chiller_6-050000::FULL
Score	TDA=1; GSR=1. The terminal action is correct and grounded in delivered evidence.
Matched-regime key	world_id:GENA-phys_in__iot_agree-chiller_6-050000; alternate conditions retain this world identifier while changing condition_id.

D.5 TEMPORAL GROUNDING: THREE-EPISODE SEQUENCE WITH A SHARED ACQUISITION BUDGET

This example tests whether the agent re-observes evidence once it may have gone stale, rather than reusing an earlier reading or exhausting its acquisition budget before the evidence is actually needed. The sequence's battery_budget is fixed at $n_episodes - 1$ by world construction (Section 5), so that re-observing at every episode is infeasible while reusing one reading elsewhere always fits giving re-observation a real, world-grounded cost rather than a free action.

E CROSS-CHECK FAILURE REVEALING A SCHEMA GAP

Figure 9 shows scenario AOBv2-REAL-018, a real-image annotation case rejected for adjudication after two independent labeling passes disagreed. The gauge carries a red *tingyong zheng* ("out-of-service certificate") marked IEA762 and dated July 15, 2021. The tag covers part of the dial, but it also communicates an operational fact: the instrument has been formally taken out of service. That status can determine the appropriate downstream action even if the remaining dial markings and needle are visually readable.



Figure 9: Out-of-service gauge.

Independent labels disagreed on gauge_readable (true versus false), gauge_value (48 versus UNREADABLE), asset (compressor versus industrial pump), and image_role (reference versus query). Nevertheless, both free-text rationales identified the same cause: the out-of-service certificate partially covers the dial and obscures numerical

Table 9: Compiled E-family example: a 3-episode sequence on metro_pump_1, battery_budget=2.

Pipeline stage	Value
Canonical world	Asset metro_pump_1; world_seed 6001; battery_budget=2 physical reads across the 3-episode sequence.
Scenario template	sample_sequence; tests re-observation of potentially stale physical evidence as the underlying reading drifts across visits.
Episode 0	value_now=1.039 (in-band [0.9,1.1]); agent reads the gauge, reports 1.043, correctly COMMITs. CC=1, CC_grounded=1; battery_reads_used=1/2.
Episode 1	value_now=1.107 (band crossed; gold:ESCALATE); agent re-observes (uses its 2nd and final budgeted read) but misreads the value as 1.092 and incorrectly COMMITs. CC=0, CC_grounded=0; battery_reads_used=2/2 (budget exhausted).
Episode 2	value_now=1.175 (further out of band; gold:ESCALATE); no budget remains, agent correctly recognizes this and reports ABORT ("battery depleted, unable to perform gauge reading") rather than fabricating a reading. CC=0, stale_state_reuse:true, reobservation_was_necessary:false (the metric does not charge the agent for a re-observation it had no budget left to make).
Executable episode	SEQ-06001 (world_seed 6001, metro_pump_1)
Score	Sequence-level: 1 of 3 episodes correctly grounded (CC_grounded: 1, 0, 0); the failure at episode 1 is a genuine misreading of freshly re-acquired evidence, not staleness, while episode 2's ABORT is the correct response to an exhausted budget rather than a temporal-grounding failure in itself.

markings. The disagreement is therefore not well described as random annotator noise. Both labelers detected a salient condition that the structured schema could not represent.

The stored annotation falls back to recommended_action: CLEAN_GAUGE, treating the obstruction as if it were dirt or another removable visual barrier. Cleaning is not equivalent to recognizing that an instrument is administratively decommissioned. This case motivates a dedicated instrument_status field, separate from gauge_readable, with values capable of representing states such as active, out of service, locked out, or status unknown. It also illustrates why cross-checking must review the semantic adequacy of the schema, rather than merely resolving labels by majority vote.

E.1 Cross-check prompt

The prompt below uses GPT-4o-mini, consistent with the primary annotation prompt's JSON contract

```
You are cross-checking another model's structured annotation
of an industrial gauge image sequence. You are given the same
image sequence and the first annotator's JSON output.
Independently judge the same fields; do not simply defer to
the first annotator.

First annotation to check:
{first_annotation_json}

Respond ONLY with this exact JSON structure, no other text:
{
  "category": "<one of: gauge_degradation | occlusion |
               scale_interpretation | glare_lighting |
               iot_contradiction | never_read | clean>",
  "gauge_readable": true | false,
  "gauge_value": "<numeric value if readable,
                  else 'UNREADABLE'>",
  "agrees_with_first_annotation": true | false,
```

```
  "disagreement_fields": ["<field name>", ...],
  "disagreement_reason": "<one sentence, or empty string
                          if agrees_with_first_annotation>"
}
```

F PERCEPTIONGAUGE IMAGE-TRANSFORMATION PIPELINE

PerceptionGauge scenes are produced through 2D Pillow compositing over base photographs, not through a 3D or vector renderer. Each of the 49 seed rows produces one reference image, one category-transformed query, and three lighting variants, giving a minimum of 245 images per completed run before optional resampling. Table 10 records the implemented transformations and parameters. A final dataset total should be reported only after an end-to-end run is frozen with its generator version, seed, per-image checksums, and category counts.



Reference Gauge degradation Occlusion

Figure 10: PerceptionGauge examples: reference, soot degradation, and occlusion.

Table 10: Implemented PerceptionGauge image transformations and parameters.

Category	Function	Mechanism	Parameters
`gauge_degradation`	`apply_gauge_degradation`	Alpha-composited fog, soot, salt, dust, or oil-film overlay; Gaussian blur; soot additionally adds pixel noise.	Overlay alpha 110–210/255; blur radius 1.2–2.5 px; noise scale $\sigma = 12$ using `rng.gauss(0,12)` and an unseeded `np.random.randn` draw.
`occlusion`	`apply_occlusion`	Opaque rectangle over a randomly selected left, right, or top third; blurred edges.	Fill RGB $(60, 55, 50)$; blur radius 1.0 px; side selected uniformly.
`scale_interpretation`	`apply_scale_interpretation`	Lanczos downscale and recomposition on a flat grey panel to simulate distant viewing.	Resize factor `rng.uniform(0.18,0.35)`; panel RGB $(130, 132, 128)$.
`glare_lighting`	`apply_glare`	Radially weighted blend with a heavily blurred copy, strongest at the frame centre.	Blur radius 4.0 px; centre-weighted mask; glare-zone intensity multiplier 1.15.
`iot_contradiction`	`apply_iot_overlay`	Top-right telemetry banner displays IoT and gauge values with a contradiction flag.	Banner box at most 220×60 px; values read from the seed row.
`never_read`	`apply_never_read`	No visual transform; represents systemic access failure rather than a visual barrier.	Base image returned unchanged.
All categories	`apply_lighting_variant`	Under-lit, standard, and over-lit post-hoc variants using `ImageEnhance`.	Brightness/contrast: $(0.55, 0.85)$, $(1.00, 1.00)$, and $(1.55, 1.20)$.

The Python `random.Random` seed controls category choices and resize factors, but `apply_gauge_degradation` also reads NumPy's global RNG for pixel noise. Consequently, `--seed` does not yet make repeated runs bit-exact. The release pipeline must seed NumPy explicitly and publish a frozen manifest before replacing the minimum-run count with a definitive total.

Mutation protection and regeneration checks. Validation should check that evidence-access changes, template substitutions, prompt rerendering, and identity-preserving mutations leave the canonical identifier unchanged; that world-state-changing mutations or crossovers produce a newly registered world; and that invalid parent links or stale gold bindings are rejected. Deterministic

regeneration should reproduce identities, variable bindings, and item provenance from the same normalized inputs, versions, and any recorded sampling seeds. Identity regression tests should include semantically equivalent normalized inputs and changes to each identity-bearing field. Report the exact fixtures, tested versions, and results before describing these invariants as implementation-verified.

F.1 AGENT-VISIBLE TOOL CATALOG

Table 11 groups representative tools by their role in the inspection episodes described in Section **??**.

Table 11: Representative agent-visible tools, grouped by their role in an episode.

Function	Representative tools	Role in an inspection trajectory
Robot state and access	`get_battery`, `list_waypoints`, `navigate_to`, `open_panel`	Determine whether the robot can safely reach and access the inspection point
Evidence acquisition	`capture_image`, `read_gauge`, `read_acoustic`, `read_iot`	Return asset-linked observations through distinct modalities
Safety and verification	`safety_gate_check`, `get_asset_state`	Check human presence, operational constraints, and asset state before action
Enterprise coordination	`get_work_order`, `check_wo_similarity`, `raise_work_order`, `commit_reading`	Reconcile evidence with maintenance state and execute the operational outcome

F.2 TOOL SURFACE: MODALITY AND OWNERSHIP

The scored episode path exposes 25 tools along two independent axes: evidence modality, which controls access, and operational ownership, the backend behind a tool.

Each tool carries one of seven fixed modality tags: physical, digital, enterprise, robot, thermal, vibration, and acoustic. `capture_image`, `read_gauge`, and `open_panel` are physical tools; `read_iot`, `get_iot_reading`, and `get_sensor_history` are digital ones. Work-order queries and terminal writes – actions that change enterprise state, such as raising a work order or committing a reading – are tagged enterprise. The remaining tags separate robot state and navigation from thermal, vibration, and acoustic sensing, each of which a matched condition can gate independently of optical capture.

The ownership axis is the backend a tool actually runs against. Robot-facing tools mirror named Spot SDK capabilities for navigation, state checks, and physical access (Boston Dynamics, 2026). IoT, vibration, work-order, forecasting, and failure-mode tools come from AssetOps-Bench. INSPECTIONBENCH adds `commit_reading`, `commit_thermal_decision`, `commit_acoustic_decision`, `escalate`, `request_observation`, and `select_capability`. The two axes cut across each other: `commit_reading` runs on the robot server yet counts as enterprise-class because it performs a terminal write, while `read_vibration` keeps its own modality even though its backend is reused. Table 11 lists the full 25-tool surface; the wider MCP-server ecosystem includes additional tools whose counts are checkout-specific and are not the number a scored episode exposes.

G ADDITIONAL RESULTS AND OPERATIONAL INSIGHTS

This appendix reports results that the main text summarizes in a sentence or omits for space, and states for each what it means for model developers and for inspection-system operators.

G.1 MATCHED EVIDENCE-ACCESS REGIMES

Figure 11 shows the 80-world A-expanded pool (240 matched episodes per model). Each world is compiled into FULL, PHYSICAL_ONLY, and DIGITAL_ONLY with the world and gold action fixed (Section 5).

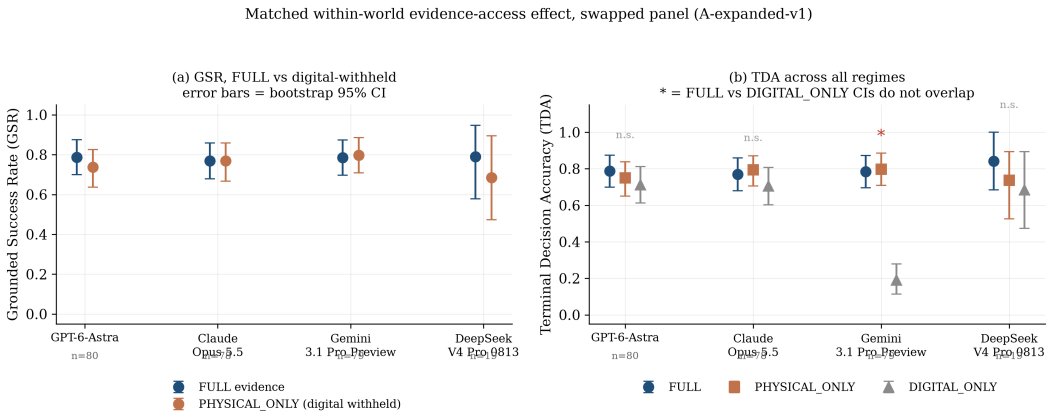


Figure 11: Matched-regime analysis. (a) GSR under FULL vs. PHYSICAL_ONLY. (b) TDA across all three regimes; $*$ marks non-overlapping FULL/DIGITAL_ONLY 95% bootstrap CIs. DeepSeek is shown at n=19 triplets.

FULL and PHYSICAL_ONLY GSR intervals overlap for all four models, so withholding digital evidence produces no distinguishable change. Withholding physical evidence separates the models. Three retain 68–71% terminal accuracy with no delivered physical observation, and their intervals overlap FULL. Gemini alone drops to 19.0% (CI 11.4–27.9), which does not overlap FULL. DeepSeek is partial (n=19); Qwen3.5-397B-A17B is omitted because its A-expanded pool is incomplete.

Model developers: DIGITAL_ONLY TDA measures how much of a correct terminal action comes from a prior rather than from the required evidence. *Practitioners:* where policy requires asset-local physical evidence, a correct decision made on digital telemetry alone should not be recorded as verification.

G.2 Physical versus enterprise decision cases in D

D is split into physical-admissibility and enterprise-workflow cases (Section 5). Table 13 compares them on terminal correctness, the only metric the two sets share.

The model ordering reverses: the two leaders on physical terminal correctness are the lowest on enterprise terminal correctness. At N=16 the intervals overlap, so we read this as a direction, not a ranking. Claude and GPT-6-Astra err only on COMMIT-gold episodes, where they escalate (5/8 and 4/8), and are perfect on the ESCALATE-gold episodes (8/8), which indicates over-caution rather than inattention. DeepSeek is correct on all 16 but executes few of the required tools. We excluded the four route-dependency episodes (0/4 for every model): a correct agent receives no successful navigation or pose confirmation and cannot reach the COMMIT gold, which is a template defect we disclose rather than score.

Model developers: physical constraint reasoning and enterprise-state reasoning are separable skills, and improving one does not imply the other. *Practitioners:* qualify a model for robot dispatch and for work-order decisions separately.

G.3 A model-generation comparison

GPT-5.2 (previous panel) and GPT-6-Astra were run on identical episode sets, so the comparison needs no cross-pool adjustment.

On A, the two models give the same verdict on 39 of 48 episodes; of the nine that differ, GPT-6-Astra corrects seven and breaks two. Improvement is uneven. Terminal accuracy and temporal consistency rise. Grounding rises less than terminal accuracy, so the TDA–GSR gap widens. Acquisition-gated success, procedural coverage, and physical-constraint recovery fall. This is one model family and one generation step, so we do not claim a trend.

Table 12: Matched-regime scores (TDA / GSR, %). DIGITAL_ONLY GSR is 0 by construction, because no physical observation can be delivered.

Model	Triplets	FULL	PHYSICAL_ONLY	DIGITAL_ONLY
GPT-6-Astra	80	78.8 / 78.8	75.0 / 73.8	71.3 / 0
Claude Opus 5.5	78	76.9 / 76.9	79.5 / 76.9	70.5 / 0
Gemini 3.1 Pro Preview	79	78.5 / 78.5	79.7 / 79.7	**19.0** / 0
DeepSeek V4 Pro 0813	19	84.2 / 78.9	73.7 / 68.4	68.4 / 0

Model developers: evaluating a new release on terminal accuracy alone would miss the regressions in B, C, and D. *Practitioners:* upgrading to a newer model can improve headline accuracy while weakening acquisition and procedure, so requalify per capability.

G.4 Over-escalation after acquisition

On the 33 episodes per model where acquisition is required and genuinely available, gold is COMMIT by construction, so this branch measures only whether a model overrides adequate acquired evidence with an escalation. It cannot test the reverse error.

Qwen's 12 failures are a strict subset of GPT-6-Astra's 27, so the failures form a shared difficulty gradient rather than unrelated per-model errors. In several episodes two models cite the identical delivered observation and reach opposite actions, which locates the failure in interpreting delivered evidence and not in acquiring it. Claude Opus 5.5 additionally returns malformed terminal responses on 18% of this branch, a separate protocol-compliance problem no other model shows here.

Model developers: acquisition selection is nearly solved (ADA/ASA ≥ 0.91), so the remaining headroom is in using acquired evidence. *Practitioners:* monitor whether an agent that has obtained the evidence it asked for still defaults to escalation, since this drives operator workload.

G.5 Procedural evaluation

The C column of Figure 5 is computed on 112 template-generated episodes (seven triggers on four assets), each prompt a parameterization of a hand-authored scenario with the decisive variable withheld. Gemini 3.1 Pro Preview satisfies ordering on 100% of episodes with coverage 0.12. Claude Opus 5.5 has the lowest ordering (54.5%) but the highest coverage (0.44). Gemini and DeepSeek execute a forbidden tool on 10% and 6.5% of episodes; the other three do not. Ordering and coverage therefore trade off across models and should be reported together.

H Operational Record Sources

Table 16 lists every record group in the replayed substrate. Counts are taken directly from the released record store, not from dataset documentation, and each provenance class in Table 17 is the sum of the rows beneath it.

Timestamps. Only 488 records carry a real capture timestamp (the REVA images' EXIF times). The remaining acoustic, thermal, and RGB records carry timestamps assigned at ingestion, and the IoT, geometry, and work-order records are unannotated. Evidence-age claims therefore rest on constructed timestamps, which is why real-timestamp staleness scenarios are not evaluated (Table **??**).

Asset coverage. Acoustic evidence exists only for `chiller_6` and `hydraulic_pump_1`, thermal evidence for `motor_01`, and work-order history for three of the four grounded assets (none for `motor_01`). The three RGB-only assets (impeller, transformer, turbine) have no executable inspection loop; they contribute asset-type diversity and are the source of the six asset types in Section 3.

Table 13: D-physical (N=70) vs. D-enterprise work-order templates (N=16, 95% CI). Enterprise recall is the fraction of required tools executed over all 20 enterprise episodes.

Model	D-physical CC	D-enterprise CC [CI]	Enterprise recall
GPT-6-Astra	0.886	0.75 [0.51, 0.90]	0.36
Claude Opus 5.5	0.871	0.69 [0.44, 0.86]	0.39
Gemini 3.1 Pro Preview	0.757	0.81 [0.57, 0.93]	0.13
DeepSeek V4 Pro 0813	0.729	1.00 [0.81, 1.00]	0.24
Qwen3.5-397B-A17B	0.614	0.81 [0.57, 0.93]	0.32

I Reproducibility, Compute, and Cost

This appendix states every parameter, cost, and hardware detail needed to reproduce the results in Section 5, and reports the project's cumulative compute honestly rather than only the number behind the final table — per FactoryBench's own convention of a dedicated compute/threshold appendix (their D.4/D.6).

I.1 Model configuration and inference parameters

All five panel models (Claude Opus 5.5, GPT-6-Astra, DeepSeek V4 Pro 0813, Gemini 3.1 Pro Preview, Qwen3.5-397B-A17B) are called through a single API gateway (TokenRouter) under an identical, frozen configuration: **temperature = 0**, **max output tokens = 2048**, no reasoning-mode override (provider default), zero-shot (no few-shot examples, no system-prompt customization per model). Retry policy is fixed at **3 attempts** per call with exponential backoff, and every failure is classified as *infrastructure* (timeout, malformed response, empty content) or *model behavior* (a well-formed but incorrect/non-compliant response) before scoring — never conflated. The evaluation protocol is versioned (`inspectionbench_v3_eval_protocol@phase8h2j.1`) and pinned against the frozen benchmark manifest's SHA-256 digest so a re-run can verify it is scoring the identical episode set.

I.2 Hardware and software infrastructure

Model inference is entirely API-mediated (TokenRouter); no local GPU or model weights are required to reproduce the agentic evaluation itself. The execution environment is a CouchDB-backed simulator (Section 4) run locally (single machine, no GPU required for the CouchDB/tool-execution layer). The one GPU-dependent stage is the PerceptionGauge annotation pipeline (Appendix B), which calls `gemini-2.5-flash` via a direct REST call from a Python 3.8 environment — chosen specifically because the standard `google-generativeai` SDK was incompatible with that environment's other dependencies, not because annotation itself requires local GPU inference. All raw per-episode records (timestamps, tool traces, scores) are persisted as JSONL, one line per episode, enabling exact reconstruction of every reported number from source records rather than from aggregated summaries alone.

I.3 Compute cost: primary reported panel

Table 18 reports measured wall-clock time for the three pools that back every number in Figure 5/6/7 (frozen-93, N=93; B-Acquisition, N=66; D-physical, N=70 — 229 episodes per model, 916 total across the four models introduced in this revision). DeepSeek V4 Pro 0813 is markedly slower than the other three on identical episode counts, consistent with the higher output-token volume observed for this model family in the legacy panel (Table I.4).

Qwen3.5-397B-A17B (retained unchanged from the prior panel; not re-run for this revision) required an additional 1.30 h on the same three pools, bringing the five-model reported-panel total to **8.81 h** of wall-clock API time.

Table 14: GPT-5.2 vs. GPT-6-Astra on identical pools.

Measure	GPT-5.2	GPT-6-Astra	Change
A terminal accuracy (TDA)	66.7	77.1	+10.4
A grounded success (GSR)	43.8	47.9	+4.1
A TDA–GSR gap (pp)	22.9	29.2	+6.3
B branched acquisition (B_AGS)	0.348	0.227	-0.121
C procedural coverage (N=17)	0.540	0.293	-0.247
D terminal correctness	0.871	0.886	+0.015
D constraint-set accuracy	0.571	0.514	-0.057
D limiting-constraint accuracy	0.571	0.464	-0.107
E temporal-consistency	0.235	0.556	+0.321

I.4 Compute cost: superseded legacy panel

The original five-model panel (Claude Sonnet 4.6, GPT-5.2, DeepSeek V4 Pro, Mistral Medium 3.5, Qwen3.5-397B-A17B), superseded in this revision by the panel above but retained here for provenance and as a completed replication of the same protocol, consumed **1,145 episodes** and **6.91 h** of measured wall-clock time across the same three pool types, plus **206,531 input / 476,540 output tokens** metered directly for the D-physical pool (the only pool whose runner captures token usage; porting that capture to the other two runners is a stated, low-cost improvement for the next revision). DeepSeek and Qwen's output-token counts were 6–8$\times$ the other three models' on identical episodes in that panel, a direct, measured signature of verbose/reasoning-heavy output rather than an inferred one.

I.5 Cumulative project scale

Across both evaluated panels, the PerceptionGauge-49 annotation pass, and this revision's validation work, the project has executed **3,061 model-episode/scenario evaluations** to date, consuming approximately **18 hours** of measured wall-clock API time (7.51 h primary panel + 6.91 h legacy panel + 2.13 h PerceptionGauge-49 annotation + $<$0.5 h validation reruns), against a constructed substrate of **4,120 canonical episodes over 1,552 grounded worlds and 21 design-level templates**, built from **116 facility-collected images**, **1,296 procedurally rendered perception scenes**, and **24,601 replayed operational records** (Section 3.1). The evaluated panel to date represents roughly **22%** of the constructed canonical pool by episode count (916+229 of 4,120 per model, before the E/A scale-up).

I.6 World Identity

The identifier used in Section 3.1 names a world, but a comparison across evidence regimes is only meaningful if two episodes actually describe the same world – their prompts and tools cannot show this on their own. We therefore hash the sampled state behind the identifier (asset, physical value, IoT value, work order and technician flags, and operating band), so two episodes share a `world_id` only when every one of these fields is identical. The compiler rejects any episode whose world has changed under regeneration, and the evaluator groups episodes by `world_id` rather than by the identifier string, so a score difference inside a matched set is attributable to what the agent could access, never to a silently different world. Two invariants are checked on every compiled episode: it must still hash to its declared `world_id`, and its gold must still hold, with world-derived gold recomputed from the hashed state and checked against the gold stored at compile time.

Table 15: Post-acquisition outcomes on the 33 acquire-and-COMMIT episodes (ADA and ASA correct).

Model	Correct	Over-escalated	Malformed
GPT-6-Astra	3 (9%)	27 (82%)	0
Claude Opus 5.5	8 (24%)	16 (48%)	6 (18%)
DeepSeek V4 Pro 0813	12 (36%)	18 (55%)	0
Gemini 3.1 Pro Preview	13 (39%)	17 (52%)	0
Qwen3.5-397B-A17B	18 (55%)	12 (36%)	0

Table 16: Sources of the 24,601 operational records. *Class*: L2 = replay from another asset of the same class; L3 = diagnostic-only; N/A = digital or enterprise record.

Modality	Source	Records	Class	Bound to asset
IoT time series	AssetOpsBench native telemetry	14,313	N/A	`metro_pump_1` 5,116; `motor_01` 4,096; `chiller_6` 2,896; `hydraulic_pump_1` 2,205
Acoustic	MIMII, pump and fan rigs (Zenodo 3384388)	9,757	L2	`chiller_6` 5,550 (fan); `hydraulic_pump_1` 4,207 (pump)
Thermal	REVA induction-motor thermal set (Mendeley pymtdhfzbj), radiometric	494	L3	`reva_induction_motor`
Thermal	Induction-motor thermal fault set (Mendeley m4sbt8hbvk), false-colour	11	L2	`motor_01`
Thermal	Rotating electromechanical rig (Dataverse 10.34810/DATA2500)	10	L3	`rotating_rig_motor`
RGB defect	Casting-product defect images (Kaggle ravirajsinh45)	2	L3	`pump_impeller_1`
RGB defect	15-class substation equipment (Zenodo 7884270)	2	L3	`transformer_substation_1`
RGB defect	Turbine blade erosion (Blade30)	2	L3	`turbine_blade_1`
Access geometry	AssetOpsBench asset-profile fixture	7	N/A	one per asset: the four grounded assets plus impeller, transformer, turbine
Work-order history	AssetOpsBench native records	3	N/A	`chiller_6`, `hydraulic_pump_1`, `metro_pump_1`
Total		**24,601**		nine assets, six asset types

Table 17: Provenance rollup. Classes are counted separately and are not comparable across rows.

Class	Records	Composition
L1 (capture from the inspected asset)	0	no record originates from the scored asset itself
L2 (same-class replay)	9,768	9,757 acoustic + 11 thermal
L3 (diagnostic-only)	510	494 REVA + 10 rotating-rig thermal + 6 RGB
Not applicable	14,323	14,313 IoT + 7 geometry + 3 work-order
Total	24,601	

Table 18: Measured wall-clock cost, primary reported panel (frozen-93 + B-Acquisition + D-physical, 229 episodes/model).

Model	frozen-93	B-Acquisition	D-physical	Total
GPT-6-Astra	26.5 min	7.7 min	6.9 min	0.69 h
Claude Opus 5.5	24.5 min	7.7 min	6.9 min	0.65 h
Gemini 3.1 Pro Preview	40.8 min	12.2 min	15.4 min	1.14 h
DeepSeek V4 Pro 0813	106.5 min	30.6 min	165.0 min	5.03 h
Total (4 models)				**7.51 h**