FROM OBSERVATION TO ACTION: STRUCTURED ENGINEERING INFERENCE FOR AGENT-DRIVEN DESIGN ON REAL CAD/CAE INFRASTRUCTURE

Anonymous authors
Paper under double-blind review

ABSTRACT

Engineering design optimization requires iterative simulation and modification, where each change must be grounded in reasoning about the simulation behavior it addresses. That cycle is exposed to software through computer-aided design and computer-aided engineering (CAD/CAE) tools, which recent tool-augmented agents can now operate end-to-end. However, operating tools is not the same as making informed design decisions. We present an engineering inference framework that structures the agent's reasoning from simulation observation to design action, with a 157-category controlled vocabulary of engineering failure modes, design modifications, and formal validation. We evaluate both the observation and the action step on real infrastructure. On observation, we extend prior benchmarks to tool-augmented and multi-view settings across 550 Finite Element Analysis questions, finding that interpretation accuracy ranges from 38-65% regardless of model tier, tool access, or viewpoints. On action, we compare an unstructured Baseline agent against an Ontology-guided agent on three self-terminating design tasks with live CAD/CAE solvers, spanning single-objective and coupled multi-disciplinary settings. On the single-objective design, the ontology agent removes 9.32% of the starting mass compared with 3.72% for the baseline agent, with all stress and displacement limits satisfied. On the coupled multi-disciplinary case, it reaches its best design at 91% of the starting mass, while the baseline finishes 23% heavier than the part it began with. The two agents differ in strategy rather than in effort: the Ontology-guided agent adds structure through ribs, while the Baseline agent adds bulk. Together, these experiments establish where in the inference loop current frontier models are effective, and where an explicit engineering vocabulary changes what the agent searches for.

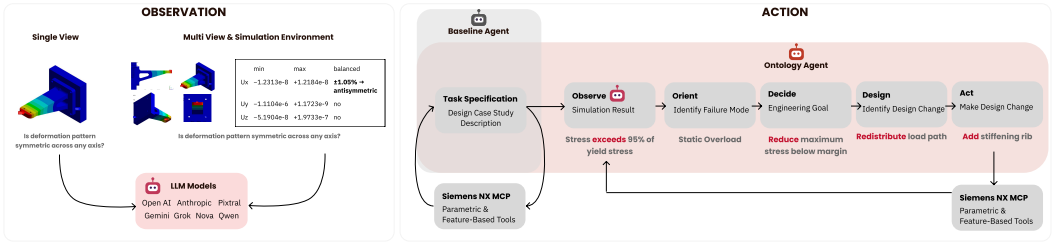

Figure 1: In observation, we test whether additional views or direct simulation data access improve how models interpret simulation results. In action, we compare a Baseline agent against an Ontology agent that routes each design change through a five-step engineering evaluation chain, both operating the same CAD/CAE tools.

1 INTRODUCTION

Engineering structures that are safe, manufacturable, and high-performing are the product of iterative cycles of design and simulation. An engineer proposes a geometry, subjects it to physics-based

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.

analysis, interprets the results, and modifies the design accordingly. This process requires more than executing a sequence of software operations. It requires engineering inference: observing what a simulation result means, orienting to the structural behavior at play, deciding on an engineering objective, identifying which physical parameters to change, and acting on that reasoning through a specific modification. When this inference is absent, design changes can make structures worse. A published aerospace design tutorial documents a case in which a human engineer reduced the mass of a helicopter pitch arm by 16.9% but increased maximum stress by 47% and deformation by 70.4%, producing what the tutorial describes as a more dangerous stress condition (Ansys, Inc., 2020). The mass reduction succeeded; the structural reasoning did not. While tool-augmented agents now automate CAD/CAE execution across multimodal parsing, tool recovery, and distributed space search (Qi et al., 2025; Zhang et al., 2026; Deng et al., 2026b; Menon et al., 2026; Humphrey et al., 2024; Zhao et al., 2026; Knapik et al., 2025), operating software remains distinct from making sound engineering choices. Current evaluations track whether pipelines execute successfully, neglecting whether generated modifications are structurally valid, or competitive with human engineering design.

For an agent to complete the engineering inference loop, it must do two things: interpret what a simulation is showing, and produce design modifications grounded in that interpretation. Neither capability has been isolated and rigorously evaluated. On interpretation, prior work has established that vision-language models struggle with engineering simulation imagery on static image inputs (Ezemba et al., 2025b;a), but these evaluations did not test whether tool access to the underlying simulation model, additional rendered viewpoints, or stronger model tiers close the gap. On action, existing agentic systems produce design changes through opaque reasoning that cannot be inspected, validated, or compared across runs, and no prior system measures agent-driven designs against documented expert designs to assess decision quality. Without evaluating both capabilities and tracing how they interact, it is not possible to determine whether the bottleneck in agent-driven design lies in how the agent interprets simulation results, how it reasons about modifications, or both.

We address this by testing interpretation and action separately. To evaluate interpretation, we extend prior benchmarks to tool-augmented and multi-view settings, measuring whether data access, additional viewpoints, or stronger models improve simulation comprehension across 550 structural finite element analysis questions. To evaluate action, we present an engineering inference ontology that structures the agent's reasoning process from observation to action. The ontology extends the Observe-Orient-Decide-Act (OODA) loop Wendt (2024) with an additional Design step, producing a five-step inference chain grounded in a 157-category controlled vocabulary with formal validation. We evaluate this ontology in controlled experiments on three structural components with live CAD/CAE solvers, comparing an unstructured **Baseline** agent against an **Ontology**-guided agent that carries the controlled vocabulary. Two case studies are single-objective and the third is a multi-coupled design. In every case the agent decides when to stop, so the iteration budget is not a design variable of the experiment. Our contributions are as follows:

1. An engineering inference ontology that structures agent reasoning from simulation observation to design action, extending the OODA loop with five typed node types, a 157-category controlled vocabulary with formal validation, and eight reference failure-mode chains. The ontology produces auditable, machine-parseable inference traces for agent-driven engineering design.

2. An empirical evaluation of the observation step across 550 FEA questions and five conditions spanning three model tiers and two tool-augmentation strategies, extending prior work (Ezemba et al., 2025b;a) to tool-augmented and multi-view settings. Accuracy ranges from 38-65% regardless of intervention, locating the observation step as a limit on agent-driven design quality.

3. A controlled comparison of Baseline and Ontology agents on three self-terminating design tasks with live CAD/CAE infrastructure, covering single-objective and coupled multi-disciplinary settings. The Ontology condition reduces mass by 9.32% compared with 3.72% for the Baseline, and on the coupled multi-disciplinary task it reaches the best design.

2 Related Works

2.1 Agentic Approaches to Simulation Automation

Recent work has applied tool-augmented foundation models to automate the end-to-end simulation workflow. Deng et al. (2026b) introduce a closed-loop agent that orchestrates CAD modeling, simulation execution, and optimization through sequential tool calls, framing the iterative design-simulation loop as a long-horizon decision-making problem under executability constraints and stochastic tool failures. Qi et al. (2025) present FeaGPT, which automates the geometry-mesh-simulation-analysis pipeline for finite element analysis, spanning geometry creation, mesh generation with automated density adaptation, boundary condition specification, solver configuration, and result interpretation. Zhang et al. (2026) extend agentic simulation to multimodal inputs with VFEAgent, enabling agents to parse raw engineering drawings and execute finite element workflows end-to-end. Menon et al. (2026) demonstrate multi-agent collaboration for automated design exploration on high-performance computing systems, where agents coordinate simulation execution and design space search across distributed resources without requiring researchers to manually construct workflow graphs or specify exploration strategies.

More recent systems have extended this trajectory. COSMO-Agent (Deng et al., 2026a) applies reinforcement learning with multi-constraint reward shaping to teach LLMs the closed-loop CAD-CAE process across 25 component categories. AbaqusAgent (Sarker et al., 2026) introduces a six-agent architecture achieving 86% pipeline success on 50 solid mechanics problems. Physics-in-the-Loop (Berger et al., 2026) embeds knowledge-based engineering tools directly into the agent's decision loop, formulating design as sequential decision-making with physical verification. However, the limitation these systems share is one of measurement rather than capability. All of them report pipeline-level success: whether the geometry was built, whether the solve completed, whether the loop closed. None reports whether the reasoning that produced a given modification can be inspected, and none measures the resulting design against a documented expert design for the same part. A system that reaches 86% pipeline success has demonstrated that it can drive the solver, not that its design decisions are comparable against a baseline. The present work addresses this specific limitation. The pipeline is fixed and only the reasoning scaffold supplied to the agent is varied. From the ontology, the agent reports the inference chain that can be read back and validated. Finally, this work reports design outcomes against both a documented expert redesign of the same component.

2.2 Simulation Interpretation and Visual Reasoning

Interpreting engineering simulation outputs remains a manual, expertise-intensive bottleneck requiring visual decoding of complex physics fields to inform design decisions (Cottrell et al., 2009; Østergård et al., 2016; Jackson et al., 2024; Dong et al., 2025; Solmaz et al., 2024; Ni & Buehler, 2024). While text-based agents automate simulation setup and configuration (Buehler, 2024; Li & Corney, 2025; Pandey et al., 2025), they leave visual interpretation unmeasured. Vision-language models excel at general visual reasoning and domain tasks like documentation and concept generation (Liang et al., 2024; Karamcheti et al., 2024; Wu et al., 2024; Doris et al., 2025; Li et al., 2025; Ghasemi & Moghaddam, 2025), yet struggle with physics-based contour fields and mesh overlays (Wang et al., 2024; Kaduri et al., 2025). Benchmarks across structural, fluid, and architectural domains show that VLMs achieve near-chance accuracy on simulation imagery, confirming a domain-specific capability gap rather than a prompt engineering issue (Ezemba et al., 2025a;b; Mehta et al., 2026).

The critical limitation of existing benchmarks is their static, image-only input regime. Whether providing tool access to underlying simulation data, additional rendered viewpoints, or programmatic model queries closes this interpretation gap remains untested; leaving open the possibility that low accuracy is an artifact of impoverished input. We address this directly by evaluating frontier models on a fixed question benchmark extended with live data tools, interactive querying, and multi-view inputs.

2.3 Design Intent and Simulation Intent Representations

Engineering decision-making has been formalized separately within CAD and simulation domains. Simulation ontologies capture pre-analysis modeling decisions to automate finite element model

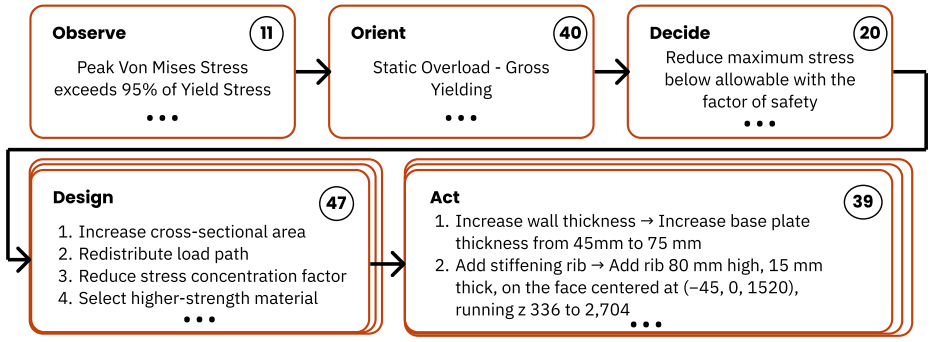

Figure 2: The engineering inference ontology encodes simulation-to-action reasoning as a five-step directed graph extending the OODA loop. Design and Act fan out one-to-many.

generation (Boussuge et al., 2019), while feature-graph representations (e.g., AFGs) encode geometric parameters and dependency edges to preserve design intent (Indupally et al., 2026). Broader engineering taxonomies demonstrate that design activities can be structured into shared vocabularies (Sim & Duffy, 2003). Yet, each framework leaves critical gaps: simulation ontologies stop upstream of analysis results, feature graphs carry no physics-based behavior, and process ontologies describe workflow categories rather than specific design inference steps.

This reflects a consistent pattern in the literature: design intent and simulation intent are formalized as separate concerns. No existing structure links what an agent observes in simulation data to what it should modify in CAD geometry. Our framework addresses this gap by establishing an integrated engineering inference structure from observation through action, validated on physical components with live CAD/CAE solvers. Ultimately, prior systems automate workflow execution but leave the engineering inference loop unmeasured; failing to evaluate whether interactive tool access aids visual interpretation, or whether structured reasoning improves agent-driven design outcomes relative to human expert baselines.

3 METHODS

3.1 ENGINEERING INFERENCE ONTOLOGY

The engineering inference ontology encodes how a simulation observation leads to an actionable design change as a five-step directed graph. The five steps extend the Observe-Orient-Decide-Act (OODA) loop with an additional DESIGN step that captures the engineering-specific task of identifying which physical parameters or features to alter. This distinction is critical for CAD/CAE execution, where design actions span both scalar parametric updates (e.g., altering a plate thickness) and generative feature operations (e.g., sketching and extruding a pocket to alter geometry). Figure 2 summarizes the five node types and illustrates the inference structure with its one-to-many fan-out at the DESIGN and ACT steps, reflecting that a single failure mode typically admits multiple independent design responses.

Each of the 157 ontology categories includes a definition, engineering significance, and a representative example. Every node type includes one "Other (Specify)" fallback category, enabling the ontology to handle cases outside the predefined vocabulary while tracking fallback frequency as a coverage metric. The schema enforces structural validity through automated validation, and a completeness metric measures spine connectivity from OBSERVE through ACT. Eight reference failure-mode chains (static overload, fatigue, fracture, buckling, corrosion, thermal failure, joint failure, and vibration) encode established engineering knowledge with cross-chain interaction edges.

3.2 EVALUATION OF INTERPRETATION

To evaluate whether tool access, additional viewpoints, or stronger models close the interpretation gap identified in prior work (Ezemba et al., 2025b;a), we extend the OpenSeeSimE-Structural-Mini

benchmark to new conditions. The benchmark contains 550 multiple choice questions about FEA contour-plot images across five structural components and three question types (binary, multiple choice, spatial). We add two conditions beyond image-only baselines: 12 read-only data tools connected to the commercial solver via an Model Context Protocol (MCP) server and 10 camera and rendering tools for additional viewpoints. In total, 17 models are evaluated from six vendors. The complete condition details, the results per-question-type and the scoring methodology are provided in Appendix C.

3.3 Evaluation of Action

This experiment evaluates how an explicit engineering ontology changes the reasoning, search behavior, and design outcomes of agent-driven modifications on real structural components with live simulation loops.

Conditions. The case studies compare two conditions that differ only in the system prompt. All conditions use the same model, tools, part, and boundary conditions.

1. **Baseline:** Task objective, tool definitions, parameter lists, and iteration protocol.
2. **Ontology:** Additionally, the 157-category ontology organized by the five-step inference chain, with the model instructed to classify each inference step into named engineering categories.

The Ontology prompt includes the category names, definitions, governing equations, worked examples, and parameter mappings for each category. The effect of each additional level of detail (intermediate levels) is given in Appendix A and Appendix H.

Case studies. Three structural components are evaluated (Figure 3): two single-objective and one multiobjective. In all cases, agents determine their own stopping point rather than exhausting an iteration budget. All runs use Claude Opus 4.6 via a model-agnostic MCP architecture (additional models in Appendix C).

The first is an unconstrained structural link loaded at 546,000 lbf at the large bore with the small bore fixed (applied load locked; Appendix F). Starting at a baseline factor of safety (FoS) of 1.0, the task is additive strengthening to reach FoS ≥ 3.0. A documented expert design serves as reference (Figure 5). This case study uses 10 random seeds per condition.

The second is a helicopter pitch arm from a published aerospace tutorial (Ansys, Inc., 2020), which documents a baseline (1.332 kg mass, 338.75 MPa peak stress, 0.381 mm deformation), and a human redesign (-16.9% mass, $+47\%$ stress)). The task requires mass minimization subject to tight constraint margins (≤ 355.06 MPa stress, ≤ 0.378 mm displacement). Unlike Case 1, this task requires feature-based topological edits (e.g., sketching, pocket extrusions, filet adjustments) rather than dimensional tuning alone. Runs operate under a safety cap of 100 iterations (unreached).

The third is a multiobjective T-bracket evaluated sequentially in Siemens NX / Simcenter 3D (infrastructure details in Appendix D). The agent optimizes three coupled physics objectives under a mass cap of 1,665.39 kg ($1.30\times$ the 1,281.07 kg baseline): minimizing peak stress (SOL 101, target 282.5 MPa, corresponding to the American Iron and Steel Institute (AISI) 1045 endurance limit at $N = 10^6$ cycles), maximizing the buckling eigenvalue (SOL 105, target 162,000), and maximizing the first natural frequency (SOL 103, target 7.5 Hz). Objectives are combined into a weighted score (weights 0.40, 0.30, 0.30). Because stiffening simultaneously alters frequency, buckling, and mass, the objectives compete directly. Isolated single-physics experiments (buckling, vibration, fatigue) are detailed in Appendix G.

Metrics. We evaluate each condition in three dimensions. *Design outcome* metrics capture the final quality: primary objective, mass and distance from the target, compared against documented expert reference designs where these exist. *Search behavior* metrics capture how the design space was covered: iterations taken, mass-budget burn rate, feasible-set hypervolume, and the fraction of a run's own peak score retained at termination. *Inference structure* metrics capture auditability: chain completeness, vocabulary fidelity, fallback rate, and the categories committed to. The token cost is decomposed into components on the prompt and completion-side.



Figure 3: Three case studies explored (a) Structural link: factor of safety targeting under static structural failure (SOL 101) (b) Helicopter pitch arm: mass minimization under stress and deformation constraints (SOL 101), with documented human engineer. (c) T-bracket: three coupled objectives on a single geometry under a shared mass constraint, combining fatigue (peak stress) (SOL 101), buckling (eigenvalue modes) (SOL 105), and vibration (first natural frequency) (SOL 103). All components were modeled in Siemens NX and solved using the integrated Simcenter 3D solver.

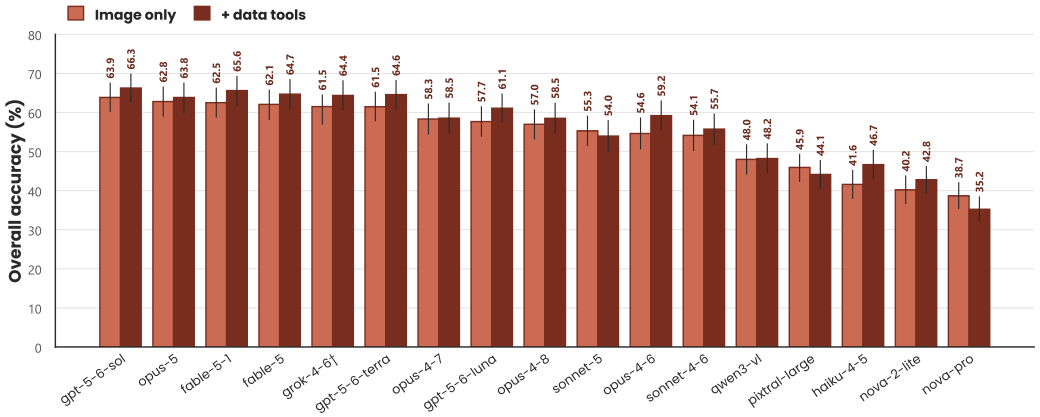

Figure 4: Simulation interpretation accuracy by model and condition. Image-only results for current models are from a seventeen-model sweep on 550 questions. Adding data and view tools yields smaller gains than moving to a stronger model, and no configuration exceeds 66.3%.

4 Results

4.1 Observation Step Evaluation

Figure 4 reports interpretation accuracy across seventeen models from six vendors on the OpenSeeSimE-Structural-Mini benchmark, evaluated over 550 questions ($N = 5$ repetitions per item) in both image-only and tool-augmented settings (12 live simulation tools connected). This extends prior image-only evaluations (Ezemba et al., 2025b;a) to tool-interactive frontier models. Image-only accuracy ranges from 38.7% (Nova Pro) to 63.9% (GPT-5.6 Sol). While tool access improves performance for 14 of the 17 models (statistically significant for 9), no configuration exceeds 66.3%. GPT-5.6 Sol achieves the marginally higher accuracy in both settings with 63.9% and 66.3% accuracy for image and tool-based analysis respectively. However, the top six models are marginally the same in both conditions, suggesting that the limitations in observations are not bridged with higher tier models. Among the 17 image-only models, multiple choice is the weakest category for 16 of them, and no model exceeds 55.3% on it, against best accuracies of 77.7% on binary and 83.6% on spatial questions. Across the 17 models, multiple-choice accuracy improves with overall accuracy at about the same rate as binary, while spatial accuracy improves about twice as fast. Stronger models are closing the spatial gap and not the multiple-choice gap. The complete per-category results are given in Appendix C.

Table 1: Single-objective design outcomes under agent-determined stopping. All runs satisfied all constraints on both parts. The structural link is run with 10 seeds per condition; the pitch arm is a single self-stopping run per condition.

			Baseline		Ontology	
Case study	Metric	N	Outcome	Iters	Outcome	Iters
Structural link	Mean FoS	10	**6.59**	1.4	6.05	1.2
Helicopter pitch arm	Mass $\Delta\%$	1	-3.72	19	$\mathbf{-9.32}$	43

4.2 Single-Objective Design Tasks

Table 1 reports the two single-objective case studies with claude-4.6 agent-determined stopping. All runs satisfy all constraints on both parts. Every run on the structural link reaches the factor of safety target within two iterations by scaling the load-bearing cross-section. The mean factor of safety is 6.59 for the Baseline and 6.05 for the Ontology condition, a difference that is not statistically significant ($p = 0.1554$). In the helicopter pitch arm, the Ontology condition reduces the mass by 9.32% (from 1.3321 kg to 1.2079 kg) compared to 3.72% for the Baseline agent (1.2825 kg). Both conditions satisfy all stress and displacement limits. The Baseline agent terminates at iteration 19, concluding that further mass reduction is impossible without violating constraints. In contrast, the Ontology agent reinterprets its trajectory as a stiffness issue at iteration 19, increasing the filet radii to gain displacement margin and using that margin to cut deeper pockets through iteration 43. See figure 6 for more details.

4.3 Coupled Multiobjective Design Task

Table 2: Coupled multi-disciplinary Design Task for Baseline and Ontology Guided Agents. Each objective is reported against the design requirement it was set to meet. Ontology Agent met 2/3 requirements against the baseline 1 out of 3 and got there with mass and time efficiency

Requirement	Failure mode	Target	Baseline Agent	Ontology Agent
Peak stress	Fatigue	$\leq$ 282.5 MPa	**232.6 (met)**	**241.4 (met)**
Buckling eigenvalue	Buckling	$\geq$ 162,000	158,900	**336,500**
First frequency	Vibration	$\geq$ 7.5 Hz	3.159	4.121
Requirements met		3	1 of 3	**2 of 3**
Cost of the design reached				
Mass at best design		$\leq$ 1,665 kg	1,579	**1,160**
Mass cap consumed		–	95%	**70%**
Search behavior and cost				
Iterations / tool calls		–	20 / 71	**19 / 66**
Input / output tokens		–	**691k / 28.7k**	1,620k / 37.7k

Table 2 summarizes the multi-coupled T-bracket task, which optimizes maximum stress, buckling eigenvalue, and natural frequency simultaneously under a mass constraint of 1,665.39 kg (1.30$\times$ the 1,281.07 kg baseline). Each objective is reported against the requirement it was set to meet. Measured against the requirements, the Ontology agent meets two of three and the Baseline agent meets one. Both met the fatigue requirement, at 232.6 MPa and 241.4 MPa compared to the endurance constraint of 282.5 MPa. Because that constraint is a threshold rather than a quantity to be minimized, the Baseline agent's larger margin adds no additional fatigue life, and it was obtained at 123% of the starting mass against 91% making it an over designed part. The Baseline agent did not meet the buckling requirement by 1.9%, but the Ontology agent exceeded it by a factor of two. Neither agent met the vibration requirement, although the Ontology agent reached 55% of the target compared to the baseline value of 42%. For search behavior, the Baseline agent consumes 90% of its mass cap by iteration 5, while the Ontology agent peaks at 77% utilization. Feasible-set hypervolume, which measures the volume of the normalized objective space dominated by a run's

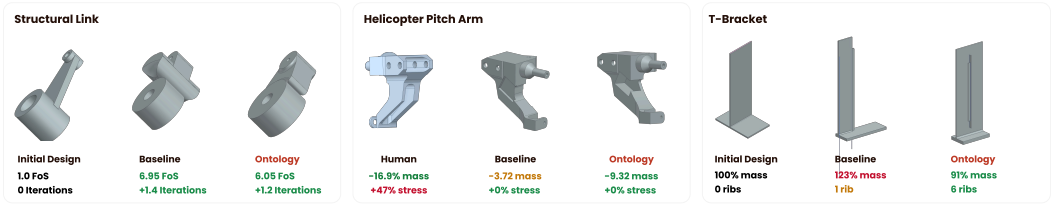

Figure 5: Final designs from each condition across the three case studies. **Structural Link:** Both agent runs reach factor of safety targets in an unconstrained, parametric driven design space. **Helicopter pitch arm:** Ontology removes 9.32% of the starting mass compared to 3.72% for Baseline, roughly half the reduction achieved by the documented human redesign but without that redesign's 47% stress increase. **T-bracket under coupled fatigue, buckling, and vibration requirements:** Ontology meets two of the three requirements at 91% of the starting mass, while Baseline meets one at 123%.

feasible designs and is therefore free of the weighting used in the combined score, is 0.260 for the Ontology agent compared to 0.046 for the Baseline. On auditability, the Ontology agent emits a complete five-step inference chain for all 19 proposals with 82% exact vocabulary fidelity and zero fallback usage.

5 Discussion

An explicit vocabulary changes what the agent searches for, not how hard it searches. On the coupled multiobjective task, the two agents made a nearly identical number of tool calls (71 and 66) over a nearly identical number of iterations (20 and 19), and their output token counts differ by less than 10k. What differs is the content of the search. The Baseline agent treated a stiffness deficit as a thickness problem and spent its mass budget accordingly. The Ontology agent, required to name the physical parameter it was targeting before naming the action, arrived at moment of inertia rather than cross-sectional area, and moment of inertia is raised more efficiently by ribs than by bulk.

Iteration budget is a confound in agent evaluation. All results in the main text use agent-determined stopping. This is a deliberate protocol choice, and the reason is visible in the preliminary bounded experiments reported in Appendix E. Under a fixed 10-iteration cap on the helicopter pitch arm, the three conditions achieve statistically indistinguishable mass reduction (-3.24%, -3.26%, -2.27%; Kruskal-Wallis $p = 0.29$, $N = 10$ per condition), and the Ontology condition's more aggressive boundary-seeking search registers as constraint violations rather than as better designs. Removing the cap changes the ordering. The cap does not scale the effect, it inverts it, because it truncates exactly the phase of the search in which the reasoning structure pays off. Evaluation protocols for design agents should therefore use agent-determined stopping or report results at multiple budget levels, and results reported under a single fixed budget should be read as budget-specific.

Auditable reasoning as a prerequisite for engineering trust. In safety-critical engineering domains, a design change must be justified and traced to the simulation behavior that motivated it. The Ontology condition produces exactly this: a typed inference graph in which each step is classified into a named engineering category, complete on every proposal and with zero fallback on the coupled task. This transforms the agent from a black box that emits a design into a process whose reasoning can be examined, validated, and compared across runs at each step. The token cost of this instrumentation is entirely prompt-side and amenable to caching, making it a practical addition to agentic design workflows rather than a prohibitive overhead.

Behavioral divergence at constraint boundaries. Outcome metrics capture where runs terminate, but reasoning traces reveal why they diverge. In both case studies, divergence occurs when the primary optimization lever hits a performance ceiling (Figure 6). On the helicopter pitch arm, mass reaches the displacement constraint. Baseline declares further mass reduction impossible and stops. Ontology treats the constraint as relaxed by secondary geometry, increasing fillet radii to gain displacement margin, which it spends on deeper pocketing. On the coupled T-bracket, with frequency

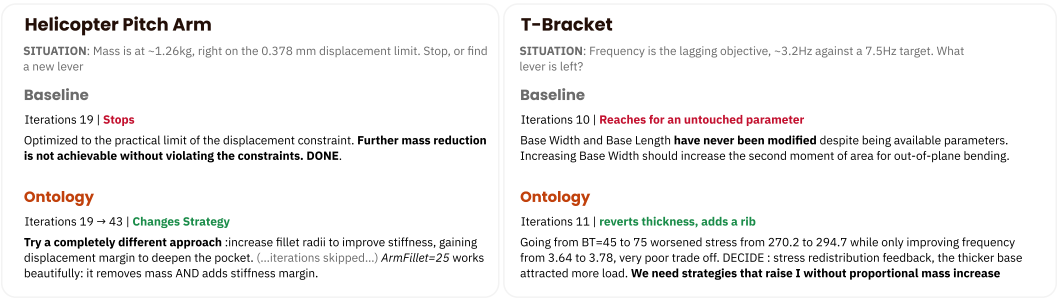

Figure 6: Decision trace divergence at critical constraint boundaries. (a) Helicopter Pitch arm task: Baseline halts upon hitting the displacement limit, while Ontology increases fillet radii to gain displacement margin for deeper pocketing. (b) Coupled T-bracket task: Baseline edits three parameters at once and loses its optimal design, while Ontology diagnoses stress redistribution from previous solve data, reverts the edit, and adds a rib. Both conditions had access to identical solver feedback.

below target and mass headroom depleted, Baseline simultaneously alters three unedited parameters, causing its score to plummet from 0.679 to 0.022. Ontology reads its previous solve data, identifies that base thickening increased stress without improving frequency, reverts the modification, and adds a rib; reaching the peak campaign score over the next four iterations. Both conditions had identical data; only Ontology synthesized past observations to pivot strategies. For additional qualitative analysis, see Appendix L.

6 Limitations

The evidence in the main text is based on a limited set of case studies. The structural link is run with 10 seeds per condition and shows no significant differences between conditions. However, the helicopter pitch arm and the coupled T-bracket are single runs per condition, so no significance test applies. The supported claim for those two is that in these runs the Ontology condition produced the best and most mass-efficient design. A expanded result exists for a bounded 10-iteration protocol on the pitch arm, described in Appendix E, where $N = 10$ per condition.

7 Conclusion

Engineering design requires interpreting what a simulation shows and acting on that interpretation through grounded feature-based and parametric modifications. We tested these two capabilities separately. For interpretation, a benchmark evaluation spanning seventeen models from six vendors, together with tool-augmented and multi-view conditions, established that no configuration exceeds 66.3% overall accuracy, which locates the observation step as a standing limit rather than a solved one. For action, we presented an engineering ontology inference framework that extends the OODA loop with a controlled 157-category ontology and evaluated it against an unstructured baseline on three components with live CAD/CAE infrastructure under agent-determined stopping. The ontology produces typed, machine-parseable inference traces, complete on every run and with zero fallback on the multi-objective coupled task. On the single-objective pitch arm, it reduces mass by 9.32% against 3.72% baseline, closing roughly half the distance to the documented expert redesign. On the multi-objective coupled T-bracket it reaches the best design at 91% of the baseline mass while the unstructured baseline agent consumes 95% of its mass budget and finishes 23% heavier. The two agents made a comparable number of tool calls over a comparable number of iterations, so the difference lies in what the agent searched for rather than in how much it searched. Making agent reasoning auditable and structured is achievable at a prompt-side cost that caching can absorb, and it changes the design strategy the agent adopts.

Ethics Statement

This work uses no human subjects and no personal data. The components studied are published tutorial geometries and internally developed test parts. We note one safety-relevant finding in the main text: a controlled vocabulary that names a mechanism for relaxing a design requirement will lead the agent to exercise it, which we observed as load reduction on the structural link task before the load was locked. We report this behavior rather than removing it from the record, because vocabulary design for engineering agents is a safety surface and the failure mode is instructive. None of the designs produced in this work were manufactured or deployed, and none should be treated as qualified for use.

Reproducibility Statement

Appendix A defines each condition, and Appendix D gives the parts, materials, meshes, loads and supports, requirements, and design parameters for every case study. Per-condition results for all design experiments are given in Appendices E through H, and benchmark details in Appendix C. The supplementary material contains the agent harness and configuration files for the helicopter pitch arm and coupled T-bracket.

AI Use Statement

Generative AI models were used in this work strictly as the subject of study: all agentic design runs and benchmark evaluations were executed using the language models described in Section 3. Generative AI was not used to formulate research hypotheses, perform primary data analysis, or generate figures or text. The authors prepared, reviewed, and verified all code, analysis, and manuscript contents, and take full responsibility for the final text and claims.

Acknowledgments

References

Ansys, Inc. Optimizing the design of a helicopter rotor pitch arm — lesson 4. Ansys Innovation Courses, FEA in Aerospace Industry, 2020. https://innovationspace.ansys.com/courses/courses/structures-in-aerospace/lessons/optimizing-the-design-of-a-helicopter-rotor-pitch-arm-lesson-4/. Accessed: 2026-09-09.

Elias Berger, Muhammad Usama, Jan Mehlstäubl, Bernhard Saske, and Kristin Paetzold-Byhain. Physics-in-the-loop: A hybrid agentic architecture for validated cad engineering design. *arXiv preprint arXiv:2605.19717*, 2026.

Flavien Boussuge, Christopher M Tierney, Harold Vilmart, Trevor T Robinson, Cecil G Armstrong, Declan C Nolan, Jean-Claude Léon, and Federico Ulliana. Capturing simulation intent in an ontology: Cad and cae integration application. *Journal of Engineering Design*, 30(10-12):688–725, 2019.

Markus J Buehler. Mechgpt, a language-based strategy for mechanics and materials modeling that connects knowledge across scales, disciplines, and modalities. *Applied Mechanics Reviews*, 76 (2):021001, 2024.

J Austin Cottrell, Thomas JR Hughes, and Yuri Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

Liyuan Deng, Shujian Deng, Yongkang Chen, Yongkang Dai, Zhihang Zhong, Linyang Li, Xiao Sun, Yilei Shi, and Huaxi Huang. Cosmo-agent: Tool-augmented agent for closed-loop optimization, simulation, and modeling orchestration. *arXiv preprint arXiv:2604.05547*, 2026a.

Liyuan Deng, Shujian Deng, Yongkang Chen, Yongkang Dai, Zhihang Zhong, Linyang Li, Xiao Sun, Yilei Shi, and Huaxi Huang. Tool-augmented agent for closed-loop optimization, simulation, and modeling orchestration. *arXiv preprint arXiv:2605.20190*, 2026b.

Zhehao Dong, Zhen Lu, and Yue Yang. Fine-tuning a large language model for automating computational fluid dynamics simulations. *Theoretical and Applied Mechanics Letters*, pp. 100594, 2025.

Anna C Doris, Daniele Grandi, Ryan Tomich, Md Ferdous Alam, Mohammadmehdi Ataei, Hyunmin Cheong, and Faez Ahmed. Designqa: A multimodal benchmark for evaluating large language models' understanding of engineering documentation. *Journal of Computing and Information Science in Engineering*, 25(2):021009, 2025.

Jessica Ezemba, Christopher McComb, and Conrad Tucker. Simulation vs. hallucination: Assessing vision-language model question answering capabilities in engineering simulations. In *Proceedings of the 7th Workshop on Design Automation for CPS and IoT*, pp. 1–9, 2025a.

Jessica Ezemba, Jason Pohl, Conrad Tucker, and Christopher McComb. Openseesime: A large-scale benchmark to assess vision-language model question answering capabilities in engineering simulations. 2025b.

Parisa Ghasemi and Mohsen Moghaddam. Vision-language models for design concept generation: An actor-critic framework. *Journal of Mechanical Design*, pp. 1–28, 2025.

LR Humphrey, AJ Dubas, LC Fletcher, and A Davis. Machine learning techniques for sequential learning engineering design optimisation. *Plasma Physics and Controlled Fusion*, 66(2):025002, 2024.

Abhishek Indupally, Ibraheem Alawadhi, Satchit Ramnath, and Jami J Shah. Bridging cad and data-driven design: Attributed feature graphs for engineering design. *arXiv preprint arXiv:2606.06405*, 2026.

Ilya Jackson, Maria Jesus Saenz, and Dmitry Ivanov. From natural language to simulations: applying ai to automate simulation modelling of logistics systems. *International Journal of Production Research*, 62(4):1434–1457, 2024.

Omri Kaduri, Shai Bagon, and Tali Dekel. What's in the image? a deep-dive into the vision of vision language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14549–14558, 2025.

Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.

Stefan Knapik, Shiguang Deng, Liwei Wang, Aaditya Chandrasekhar, Wing Kam Liu, Jian Cao, and Wei Chen. Co-design of geometry and thermal-elastic gradient alloy distribution with temperature-dependent material properties: S. knapik et al. *Structural and Multidisciplinary Optimization*, 68(7):127, 2025.

Beichen Li, Rundi Wu, Armando Solar-Lezama, Changxi Zheng, Liang Shi, Bernd Bickel, and Wojciech Matusik. Vlmaterial: Procedural material generation with large vision-language models. *arXiv preprint arXiv:2501.18623*, 2025.

Shuang Li and Jonathan Corney. Mechrag: a multimodal large language model for mechanical engineering. *Communications Engineering*, 4(1):187, 2025.

Zijing Liang, Yanjie Xu, Yifan Hong, Penghui Shang, Qi Wang, Qiang Fu, and Ke Liu. A survey of multimodel large language models. In *Proceedings of the 3rd International Conference on Computer, Artificial Intelligence and Control Engineering*, pp. 405–409, 2024.

Manasi Dushyant Mehta, Jessica Ezemba, Conrad Tucker, and Christopher McComb. Vizarqa: A foundation for visual question answering in architectural simulation. *ASME Journal of Engineering for Sustainable Buildings and Cities*, 7(2):021002, 2026.

Harshitha Menon, Charles F Jekel, Kevin Korner, Brian Gunnarson, Nathan K Brown, Michael Stees, M Giselle Fernandez-Godino, Walter Nissen, Meir H Shachar, Dane M Sterbentz, et al. Multi-agent collaboration for automated design exploration on high performance computing systems. *arXiv preprint arXiv:2603.11515*, 2026.

Bo Ni and Markus J Buehler. Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. *Extreme Mechanics Letters*, 67:102131, 2024.

Sandeep Pandey, Ran Xu, Wenkang Wang, and Xu Chu. Openfoamgpt: A retrieval-augmented large language model (llm) agent for openfoam-based computational fluid dynamics. *Physics of Fluids*, 37(3), 2025.

Yupeng Qi, Ran Xu, and Xu Chu. Feagpt: an end-to-end agentic-ai for finite element analysis. *arXiv preprint arXiv:2510.21993*, 2025.

Titu Ranjan Sarker, Muhammed Jawaad Zulqernine, Ling Yue, Shaowu Pan, Chenxi Wang, and Shiyao Lin. A multi-ai-agent framework enabling end-to-end finite element analysis for solid mechanics problems. *arXiv preprint arXiv:2606.00138*, 2026.

Siang Kok Sim and Alex HB Duffy. Towards an ontology of generic engineering design activities. *Research in Engineering Design*, 14(4):200–223, 2003.

Serkan Solmaz, Liesbeth Kester, and Tom Van Gerven. An immersive virtual reality learning environment with cfd simulations: Unveiling the virtual garage concept. *Education and Information Technologies*, 29(2):1455–1488, 2024.

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Song XiXuan, et al. Cogvlm: Visual expert for pretrained language models. *Advances in Neural Information Processing Systems*, 37:121475–121499, 2024.

Donnie W Wendt. The ooda loop. In *The Cybersecurity Trinity: Artificial Intelligence, Automation, and Active Cyber Defense*, pp. 203–222. Springer, 2024.

Jiannan Wu, Muyan Zhong, Sen Xing, Zeqiang Lai, Zhaoyang Liu, Zhe Chen, Wenhai Wang, Xizhou Zhu, Lewei Lu, Tong Lu, et al. Visionllm v2: An end-to-end generalist multimodal large language model for hundreds of vision-language tasks. *Advances in Neural Information Processing Systems*, 37:69925–69975, 2024.

Jiachen Zhang, Junyi Lao, Chenghao Liu, Siyuan Liu, Shixin Wu, Linsen Zhang, Boyu Wang, and Songfang Huang. Vfeagent: A multimodal agent framework for end-to-end automated finite element analysis. *arXiv preprint arXiv:2605.28978*, 2026.

Zhenyuan Zhao, Yu Xing, Tianyang Xue, Lingxin Cao, Xin Yan, and Lin Lu. Automs: Multi-agent evolutionary search for cross-physics inverse microstructure design. *arXiv preprint arXiv:2603.27195*, 2026.

Torben Østergård, Rasmus L. Jensen, and Steffen E. Maagaard. Building simulations supporting decision making in early design – a review. *Renewable and Sustainable Energy Reviews*, 61: 187–201, 2016. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2016.03.045. URL `https://www.sciencedirect.com/science/article/pii/S136403211600280X`.

A Condition Definitions

All conditions differ only in the system prompt. Model, tools, part, mesh, boundary conditions, and stopping rule are held fixed within each experiment.

The main text reports Baseline and Ontology. Case studies 1 and 2 use the vocabulary variant of Ontology; case study 3 uses the full variant. The Information condition and the two intermediate Ontology variants are reported in Appendix H.

Table 3: Mapping between the condition names used in the paper and the configuration names used in the experiment harness and released records.

Paper name	Configuration name	Prompt adds
Baseline	`bare`	Task, tool definitions, parameter list, iteration protocol
Information	`language`	+ the five-step inference chain, unlabeled
Ontology (vocabulary)	`language_taxonomy`	+ the 157-category controlled vocabulary
Ontology (coupled)	`multicoupled_taxonomy`	+ coupling specification and `coupling_lookup` tool
Ontology (full)	`full_taxonomy`	+ equations, examples, parameter mappings, all tools

Infrastructure. All parts are modeled in Siemens NX and solved with Simcenter 3D. The helicopter pitch arm and structural link use 3 mm CTETRA(10) elements with SOL 101 (linear statics). The T-bracket uses 120 mm elements with SOL 101, SOL 103, and SOL 105 solved in sequence at each iteration. Constraint limits are set at $1.05\times$ the respective NX baseline values to account for mesh discretization variability. The experiment harness communicates with a live NX session through an MCP server exposing 34 CAD/CAE tools, including geometry-creation tools for ribs, fillets, and pockets on the coupled task. Per-iteration timing is approximately 2-3 minutes, dominated by the NASTRAN solve. The part is reset to baseline at the start of each run. Full parameter lists, boundary conditions, and the complete tool inventory are provided in Appendix D

B Taxonomy Specification

The taxonomy is implemented as five node types, named Observe, Orient, Decide, Design, and Act. Category counts are 11, 40, 20, 47, and 39 respectively, totaling 157. An example of this chain can be found in Figure 2 and the breakdown is in Table 4.

Table 4: Inference framework node types and taxonomy category counts.

Step	Node Type	Categories	Role
1	Observe	11	Simulation result
2	Orient	40	Failure mode
3	Decide	20	Engineering goal
4	Design	47	Physical parameter
5	Act	39	Design change
	Total	157	

C Interpretation Benchmark Details

Benchmark and sweep. The benchmark is OpenSeeSimE-Structural-Mini (Ezemba et al., 2025b), a set of multiple-choice questions over FEA contour-plot images. The sweep reported below covers 550 questions: 165 binary, 275 multiple choice, and 110 spatial. All seventeen models were run through the same provider API and answered every question five times in each of two conditions: from the image alone, and with 12 tools connected to a live Ansys Mechanical session holding the simulation model behind the image.

Accuracy across seventeen models. Table 5 gives the full sweep, with accuracy pooled over the five passes. Tools raise accuracy for 14 of the 17 models and significantly for 9, with the largest gains for Haiku 4.5 ($+5.1$ points) and Opus 4.6 ($+4.6$). The six highest-scoring models, GPT-5.6 Sol, Opus 5, Fable 5.1, Fable 5, GPT-5.6 Terra and Grok 4.6, are statistically indistinguishable from one another in both conditions (paired bootstrap over questions, Holm-corrected).

Which question types respond to model scale. Table 6 compares each category with the overall accuracy of the 17 models. Multiple choice improves with model tier at essentially the same rate as binary. Spatial questions improve about twice as fast, and the highest-scoring model in the sweep

Table 5: Simulation interpretation accuracy (%) across seventeen models and six vendors, from the image alone and with 12 live data tools, on 550 FEA contour-plot questions (165 binary, 275 multiple choice, 110 spatial) answered five times each. Δ is the change from adding tools; * marks a change whose 95% interval excludes zero. †Grok 4.6 ran with a larger turn budget in the tools condition (16 turns and 10 tool calls, against 8 and 12).

		Image only				With data tools				
Model	Vendor	Overall	Binary	MC	Spatial	Overall	Binary	MC	Spatial	Δ
gpt-5-6-sol	OpenAI	**63.9**	**77.7**	47.6	**83.6**	**66.3**	81.2	50.7	**82.7**	+2.4*
opus-5	Anthropic	62.8	73.0	52.7	72.9	63.8	74.8	54.8	70.0	+1.0
fable-5-1	Anthropic	62.5	73.9	51.1	73.8	65.6	77.6	54.6	75.1	+3.1*
fable-5	Anthropic	62.1	76.0	47.3	78.2	64.7	**81.6**	50.0	76.2	+2.6*
grok-4-6†	xAI	61.5	72.6	**55.3**	60.0	64.4	78.3	**57.6**	60.4	+2.9*
gpt-5-6-terra	OpenAI	61.5	71.5	48.1	79.6	64.6	77.0	51.4	78.9	+3.1*
opus-4-7	Anthropic	58.3	72.8	51.6	53.3	58.5	70.3	55.0	49.8	+0.2
gpt-5-6-luna	OpenAI	57.7	65.6	46.2	74.5	61.1	72.8	49.5	72.5	+3.4*
opus-4-8	Anthropic	57.0	65.2	52.7	55.3	58.5	65.5	55.4	55.8	+1.5
sonnet-5	Anthropic	55.3	65.5	50.5	51.8	54.0	65.1	49.5	48.4	−1.3
opus-4-6	Anthropic	54.6	70.5	49.9	42.5	59.2	80.1	54.8	38.7	+4.6*
sonnet-4-6	Anthropic	54.1	66.8	48.2	50.0	55.7	66.8	51.5	49.8	+1.6
qwen3-vl	Qwen	48.0	53.9	45.3	45.8	48.2	57.7	43.6	45.5	+0.2
pixtral-large	Mistral	45.9	66.1	35.0	43.1	44.1	64.0	34.7	38.0	−1.8
haiku-4-5	Anthropic	41.6	58.1	31.8	41.5	46.7	54.3	47.6	32.7	+5.1*
nova-2-lite	Amazon	40.2	54.5	31.6	40.2	42.8	53.7	39.6	34.4	+2.6*
nova-pro	Amazon	38.7	57.3	30.4	31.3	35.2	52.4	26.3	32.0	−3.4*
Best		**63.9**	**77.7**	**55.3**	**83.6**	**66.3**	**81.6**	**57.6**	**82.7**	
Worst		38.7	53.9	30.4	31.3	35.2	52.4	26.3	32.0	

(`gpt-5-6-sol`) reaches 83.6% on them. Spatial reasoning is the weakest category for only one of the seventeen models, `opus-4-6`, while `gpt-5-6-sol` reaches only 47.6% on multiple choice. This is in contrast to previous evaluations, where spatial reasoning was identified as the limiting category (Ezemba et al., 2025b); those evaluations were conducted on older models, and across the wider sweep current models show that the limiting category is multiple choice. The same pattern holds with tools, where spatial accuracy again rises fastest.

Table 6: Per-category accuracy against overall accuracy across the seventeen image-only models, with the mean accuracy of the six strongest and six weakest models. Slope is the change in category accuracy per point of overall accuracy.

Category	Slope	r	Range	Top 6	Bottom 6	Gain
Binary	0.783	0.890	53.9-77.7%	74.1%	59.5%	14.7
Multiple choice	0.855	0.890	30.4-55.3%	50.4%	37.0%	13.3
Spatial	1.686	0.867	31.3-83.6%	74.7%	42.0%	32.7
Overall	–	–	38.7-63.9%	62.4%	44.8%	17.6

Consistency across passes. Answers are stable from one pass to the next. On average, 80.9% to 97.2% of a question's five answers match its most common answer, depending on the model and condition, and every model above 50% accuracy exceeds 92%. GPT-5.6 Sol gives the same answer in all five passes on 80% of questions. The errors that remain therefore recur across passes rather than varying at random: GPT-5.6 Sol answers 27.6% of questions wrongly in every pass, in both conditions.

Cost and latency. Table 7 reports the operational characteristics of each model. Median s/q is the median wall-clock time to answer one question, and token counts are for one full pass over the 550 questions. Adding tools raises input tokens, because each tool result is returned to the model. `grok-4-6` produces 1.5 to 1.7 million output tokens per pass, against at most 300 thousand for

every other model, because it reasons at length before answering. From its accuracy, the additional generation does not translate into higher interpretation accuracy on this benchmark.

Table 7: Operational characteristics per model. Median s/q is the median wall-clock seconds per question. Token counts are in thousands per full pass over the 550 questions. Tools/q is the mean number of tool calls per question.

	Image only			With data tools			
Model	s/q	In (k)	Out (k)	s/q	In (k)	Out (k)	Tools/q
gpt-5-6-sol	4.1	681	84	4.6	1,057	164	1.39
opus-5	7.4	754	153	7.3	4,616	219	1.59
fable-5-1	5.6	753	60	10.8	4,095	195	0.99
fable-5	9.1	753	138	12.1	4,622	254	1.41
grok-4-6	113.8	344	1,469	50.2	2,963	1,685	7.78
gpt-5-6-terra	3.3	682	90	5.5	1,090	179	1.88
opus-4-7	3.3	755	44	5.9	2,934	89	0.39
gpt-5-6-luna	3.4	682	113	4.7	1,050	178	1.56
opus-4-8	3.4	752	52	3.9	2,491	69	0.33
sonnet-5	4.4	753	72	6.2	2,105	75	0.14
opus-4-6	5.1	681	50	12.4	6,717	295	2.73
sonnet-4-6	4.0	681	65	9.1	6,248	272	2.55
qwen3-vl	2.3	632	38	2.5	1,344	46	0.03
pixtral-large	41.3	1,251	33	2.8	2,466	57	0.23
haiku-4-5	3.7	681	60	9.3	6,913	266	2.94
nova-2-lite	2.0	198	32	4.0	3,026	134	1.06
nova-pro	1.8	1,173	20	2.0	1,841	25	0.00

Breakdown by question type. Table 8 gives the change in accuracy from adding tools for each question type. Tools raise multiple-choice accuracy for 13 of the 17 models and binary accuracy for 10, while spatial accuracy rises for only 4 models and by at most 1.3 points. This suggests that spatial accuracy is driven by the model rather than by tool access.

Table 8: Change in accuracy (percentage points) from adding live data tools, by question type. Positive values mean tools helped. †Grok 4.6 ran with a larger turn budget in the tools condition.

Model	Binary	MC	Spatial	Overall
gpt-5-6-sol	+3.5	+3.1	−0.9	+2.4
opus-5	+1.8	+2.2	−2.9	+1.0
fable-5-1	+3.6	+3.5	+1.3	+3.1
fable-5	+5.6	+2.7	−2.0	+2.6
grok-4-6†	+5.7	+2.3	+0.4	+2.9
gpt-5-6-terra	+5.5	+3.3	−0.7	+3.1
opus-4-7	−2.5	+3.3	−3.5	+0.2
gpt-5-6-luna	+7.3	+3.3	−2.0	+3.4
opus-4-8	+0.2	+2.7	+0.5	+1.5
sonnet-5	−0.4	−1.0	−3.5	−1.3
opus-4-6	+9.6	+4.9	−3.8	+4.6
sonnet-4-6	0.0	+3.3	−0.2	+1.6
qwen3-vl	+3.8	−1.7	−0.4	+0.2
pixtral-large	−2.1	−0.3	−5.1	−1.8
haiku-4-5	−3.8	+15.9	−8.7	+5.1
nova-2-lite	−0.8	+8.0	−5.8	+2.6
nova-pro	−5.0	−4.1	+0.7	−3.4
Models improved	10 of 17	13 of 17	4 of 17	14 of 17

Caveats. Two considerations apply to the sweep. First, running all models through a single provider API standardizes prompt handling, although results may differ on first-party APIs. Second, Grok 4.6 ran with a larger turn budget in the tools condition, so its tools result is not directly comparable with the other sixteen models.

D EXPERIMENTAL INFRASTRUCTURE

All parts are modeled in Siemens NX and solved with the NX Nastran solver in Simcenter 3D. An experiment harness drives a live NX session through a Model Context Protocol (MCP) server, so every design change the agent proposes is built in CAD, meshed, and solved before the agent sees the result. Table 9 gives the part, material, mesh, loads and supports for each case study, and Table 10 gives the requirements and design parameters.

Table 9: Parts, materials, meshes, loads and supports. The helicopter pitch arm and structural link use ten-node tetrahedral elements (CTETRA(10)).

Case study	Units	Material	Element size	Loads, supports and solutions
Helicopter pitch arm	mm, N	Steel	3 mm	SOL 101. Force $(0, -1000, 400)$ N on the shaft face and $(1000, 0, -400)$ N on the bolt-hole faces; three faces fixed.
Structural link	inch, lbf	Aluminum 6061-T6, yield 35,000 psi	automatic	SOL 101. 546,000 lbf in $-z$ at the large bore; small bore fixed in all six degrees of freedom.
T-bracket (coupled)	mm, N	AISI 1045 for SOL 101 ($E = 205$ GPa, $\nu = 0.29$, yield 310 MPa, ultimate 565 MPa); steel for SOL 103 and 105	120 mm	Base face fixed in all three solutions. SOL 101: 6 MN in $-z$ on the top face. SOL 105: unit load in $-z$ on the same face. SOL 103: no load. The three solutions run in sequence at every iteration.

Table 10: Requirements and design parameters.

Case study	Requirements	Settable parameters
Helicopter pitch arm	Minimize mass from 1.332 kg, with peak stress $\leq$ 355.06 MPa and peak displacement $\leq$ 0.378 mm. Both limits are 1.05 times the starting design's values of 338.15 MPa and 0.360 mm, to allow for mesh discretization.	Four fillet radii (arm, arm base, base, front; starting at 15, 8, 7 and 5 mm), plus pocket, fillet and rib features. Thirty-one interface dimensions are frozen, covering the overall envelope, shaft, bore, nozzle and existing pocket; a change to any of them is rejected.
Structural link	Factor of safety $\geq$ 3.0, where factor of safety is yield stress over peak von Mises stress. The starting design has a factor of safety of 1.0.	Large and small bore diameters and holes, height, small-end height, thickness, fillet radius and beam length. The load cannot be changed.
T-bracket (coupled)	Peak stress $\leq$ 282.5 MPa from 300 MPa (the AISI 1045 endurance limit at 10^6 cycles); first buckling eigenvalue $\geq$ 162,000 from 135,900; first natural frequency $\geq$ 7.5 Hz from 2.93 Hz. Mass $\leq$ 1,665.39 kg, 1.30 times the starting 1,281.07 kg; a heavier design is infeasible.	Base length [500, 3000], base thickness [10, 300], base width [500, 2000], fillet size [5, 400] and T-flange start position [100, 1200], all in mm. Values outside these bounds are not applied. T-flange length and thickness are frozen. Feature operations: rib, pocket, edge fillet, sketch with extrude or revolve, pattern, mirror, and boolean.

Agent interface. On the helicopter pitch arm and the structural link, the agent calls NX tools directly. The main tool applies a set of parameter values, rebuilds the part, remeshes, reapplies the loads and supports, and solves in a single call. Further tools read and set design parameters, report solve status, create pocket, fillet and rib features, describe faces, and export result images. On the coupled T-bracket, the agent submits each design change through a single proposal tool that carries the parameter changes, any feature operations, and, in the Ontology conditions, the five-step

classification of the change. Each proposal triggers the three solutions in sequence. The agent can also query a memory of earlier iterations, inspect the current geometry, list and delete features, and reset the part to its starting geometry. The Ontology conditions additionally have tools for searching and navigating the controlled vocabulary.

Supports and loads after geometry changes. Face identifiers in NX change when the geometry is rebuilt. Loads and supports are therefore reapplied after every change by matching faces on stable geometric properties, such as surface type, area, centroid and position on the part, rather than by face index.

Timing. On the coupled T-bracket, the three solutions take a median of 2.0 minutes per iteration (interquartile range 1.7 to 2.4 minutes, over 101 iterations in five runs). On the structural link, one iteration takes a median of 90 seconds, including the agent's turn. On the helicopter pitch arm, the Baseline and Ontology runs average 52 and 75 seconds per iteration, including the agent's turn.

E Bounded Campaign: Statistically Powered Results

This appendix details the preliminary evaluation campaign, which evaluated 10 random seeds per condition under a fixed 10-iteration cap. As the only campaign in this study structured with a minimum sample size for statistical significance testing, its main finding is that under a fixed budget, all conditions produce statistically indistinguishable design outcomes.

Table 11: Helicopter pitch arm, bounded protocol, 10-iteration cap, $N = 10$ seeds per condition. Mass reduction is not significantly different across conditions (Kruskal-Wallis $H = 2.496$, $p = 0.2872$). Token cost is ($H = 20.648$, $p = 3.3 \times 10^{-5}$).

Condition	N	Constraint sat.	Mass Δ% (mean$\pm$sd)	Params changed	Tokens (mean$\pm$sd)
Baseline	10	60%	-3.24 ± 2.1	16.5 ± 2.6	$156{,}895 \pm 19{,}648$
Information	10	**90%**	-3.26 ± 1.1	16.9 ± 3.2	$181{,}552 \pm 22{,}704$
Ontology	10	70%	-2.27 ± 1.1	15.5 ± 3.5	$230{,}030 \pm 19{,}188$

F Vocabulary-Induced Load Reduction and the Force-Locked Re-Run

The structural link task is run under two parameterizations. Under the unlocked parameterization, `force_magnitude` and `force_direction` are exposed to the agent through `nx_update_geometry_and_solve` alongside the seven geometric parameters. Under the locked parameterization, both are removed from the tool schema, the load is pinned to the configuration default of 546,000 lbf, and the task instructions state that the applied load is fixed and cannot be changed. The system prompts are identical under both, so the 157-category vocabulary and its *Reduce applied demand* category remain available to the Ontology condition in either case. All structural link results reported in the main text use the locked parameterization. Factor of safety is yield stress divided by peak von Mises stress. Lowering the applied load raises that ratio without modifying the part, so a run that reduces the load can reach the target of 3.0 without strengthening the geometry. A run is recorded as a load reduction if its final `force_magnitude` is below 546,000 lbf.

Load reduction rates. Table 12 reports the unlocked parameterization. Ontology reduces the load in five of ten runs, against one of ten for Baseline and one of ten for Information. In the five Ontology runs the load is cut to between 30,000 and 100,000 lbf, a reduction of between five and eighteen times, and the reported factor of safety in those runs ranges from 9.44 to 29.92 against a target of 3.0. This demonstrates that the ontology condition explored decision nodes recommending alternative design strategies, such as load reduction, rather than the naive parameter selections made by the Baseline and Information conditions. However, because this task was intended to be geometry-driven, subsequent runs locked force modifications to render those load-reduction branches unreachable.

Table 12: Structural link, unlocked parameterization, $N = 10$ seeds per condition. A load reduction is a run whose final `force_magnitude` is below 546,000 lbf. The final column restricts the sample to runs in which the load is unchanged.

Condition	N	Load reduced	FoS (mean±sd)	Median FoS	FoS, load unchanged
Baseline	10	1/10	5.15 ± 3.4	4.29	4.12 ± 0.88 ($n = 9$)
Information	10	1/10	5.22 ± 2.8	4.44	4.41 ± 1.05 ($n = 9$)
Ontology	10	**5/10**	12.18 ± 9.6	7.23	4.39 ± 0.70 ($n = 5$)

Effect on the reported metric. Across all unlocked runs, Ontology reaches a mean factor of safety of 12.18 against 5.15 for Baseline, and the difference is significant (Mann-Whitney $U = 21.0$, $p = 0.0311$, Cliff's $\delta = -0.58$). Restricted to runs in which the load is unchanged, the three conditions reach 4.12, 4.41 and 4.39, and no pairwise comparison is significant. The separation is produced by the runs that alter the load rather than by the designs.

Condition specificity. Information receives the same five-step inference chain as Ontology and differs only in carrying the 157-category vocabulary, and it reduces the load at the Baseline rate. The behavior is confined to the condition that names the action. Reasoning traces in the five Ontology runs cite the vocabulary category directly:

> **DESIGN** (Increase cross-sectional area + Increase fillet/blend radius + **Reduce applied demand**): ... Reduce force from 546,000 to ∼45,500lbf (÷12 reduction)

Results under the force lock. Table 13 reports the locked parameterization over the same ten seeds per condition. No run in any condition reduces the load, and all thirty runs reach the factor of safety target. The conditions are not distinguishable on design outcome (Kruskal-Wallis $H = 3.724$, $p = 0.1554$), on iteration count ($H = 0.514$, $p = 0.7733$), or on parameters changed ($H = 1.038$, $p = 0.5951$). Token cost is separated ($H = 8.000$, $p = 0.0183$), with Ontology spending 46% more than Baseline and 64% more than Information.

Table 13: Structural link, force-locked re-run, $N = 10$ seeds per condition. Zero load reductions across all 30 runs.

Condition	N	Success	FoS (mean±sd)	Median FoS	Tokens (mean±sd)
Baseline	10	100%	6.59 ± 2.4	5.42	$9{,}584 \pm 5{,}844$
Information	10	100%	5.06 ± 1.0	4.97	$8{,}537 \pm 3{,}704$
Ontology	10	100%	6.05 ± 1.2	6.43	$13{,}985 \pm 5{,}669$

G Single-Failure-Mode Experiments

Three experiments isolate one failure mode each on the same T-bracket, so that the coupled task reported in the main text can be read against the uncoupled cases. Each uses one seed per condition. Buckling maximizes the first eigenvalue from a baseline of 45,650 and has no upper target. Vibration raises the first natural frequency from a baseline of 2.93 Hz to a target of 7.5 Hz, a 1.5 times margin over a 5 Hz operating frequency. Fatigue reduces peak von Mises stress from a baseline of 503.3 MPa to the 250 MPa endurance limit. Material and applied load are frozen in all three, so no result is reachable by relaxing the problem. Table 14 shows that the reported values across these conditions.

Buckling. The conditions diverge by more than an order of magnitude, ranging from 185,100 for Baseline to 2,327,000 for Ontology. The ontology condition reached the best design by exploring a different design strategy of strengthening the Flange and not the base.

Vibration. All three conditions reach the target of 7.5 Hz with all frequency final values within 1.4% of one another. The conditions differ instead in the mass spent: against a 1,281.07 kg baseline,

Table 14: Single-failure-mode T-bracket experiments, one seed per condition, agent-determined stopping. No condition met Fatigue Requirement of 250 MPa but for conditions that passed (Buckling and Vibration) Ontology condition had largely highest value and least mass respectively.

Failure mode	Metric	Baseline	Information	Ontology
Buckling (SOL 105)	Eigenvalue $\Delta\%$	$+305$	$+2{,}104$	$\mathbf{+4{,}997}$
	Iterations	50	50	69
Vibration (SOL 103)	Frequency (Hz)	**7.617**	7.513	7.565
	Mass (kg)	2,220.3	2,401.9	**2,081.6**
	Iterations	35	44	58
Fatigue (SOL 101)	Peak stress (MPa)	417.6	**362.2**	392.0
	Mass (kg)	13,079	**7,122**	12,840
	Iterations	50	81	**71**

Ontology reaches the target at $+62.5\%$, Baseline at $+73.3\%$ and Information at $+87.5\%$. Raising the first mode from 2.93 to above 7.5 Hz is a 2.6 times increase and requires stiffening that adds mass in every condition.

Fatigue. No condition successfully achieved the 250 MPa stress target. Peak stress was reduced from 503.3 MPa to 417.6 MPa (Baseline), 362.2 MPa (Information), and 392.0 MPa (Ontology), yielding safety factors of 0.599, 0.690, and 0.638, respectively, against the required 1.0 threshold. Although Information achieved the lowest peak stress at roughly half the mass of the other conditions, both Information and Ontology correctly identified the target as unreachable and self-terminated. Notably, Ontology recognized unfeasibility earlier than information, conserving significant iteration budget, whereas Information spent excess iterations attempting further optimization. Crucially, while Information appears superior on scalar stress reduction, all three conditions failed the constraint; a binary outcome in engineering where near-misses remain non-compliant.

Navigating Competing Design Objectives. Because each cell represents a single run, the ranking of conditions in Table 14 is not statistically significant. However, the clear advantage of the ontology condition in the multi-objective setting (Table 2) suggests that ontology-driven guidance becomes particularly beneficial when managing simultaneous, competing objectives; better reflecting real-world engineering design where trade-offs must be navigated.

H Multiobjective Design on T Bracket Task

This section reports all five conditions of Section 4.3 on the coupled T-bracket task, which the main text covers only for Baseline and Ontology (full). The task minimizes peak stress, maximizes the first buckling eigenvalue and maximizes the first natural frequency at once, under a hard mass cap of 1,665.39 kg, which is 1.30 times the 1,281.07 kg baseline. Five geometric parameters are configurable. Each condition is run once with a 20-iteration cap.

Combined score. Each objective is normalized from its baseline to its target and clipped to $[0, 1]$, then weighted: stress 0.40 (300 to 282.5 MPa, the AISI 1045 endurance limit at $N = 10^6$), eigenvalue 0.30 (135,900 to 162,000), frequency 0.30 (2.93 to 7.5 Hz). *Peak score* is the highest combined score reached in a run and *@Iter* the iteration at which it occurred. *Retained* is the final score as a fraction of that run's own peak, so it measures whether a condition still holds its best design when the run ends. *% of cap* is mass at the peak design as a fraction of the mass limit.

Score is non-monotonic across ontology conditions. Adding the vocabulary alone reduces the peak score to 0.440, below both Baseline (0.679) and Information (0.759), and adding the coupling specification yields negligible recovery at 0.457. Only the full condition, which supplies governing equations, examples, and parameter mappings alongside the vocabulary, outperforms Information. The two intermediate conditions also peak early at iterations 2 and 4, demonstrating that while

Table 15: All five conditions on the coupled T-bracket task, one run per condition.

Condition	Peak score	@Iter	Iters	Mass at best (kg)	% of cap	Ribs	Retained
Baseline	0.679	6	20	1,579	95%	1	0.2%
Information	0.759	6	19	1,593	96%	2	0.5%
Ontology (vocabulary)	0.440	2	19	1,570	94%	0	0.7%
Ontology (coupled)	0.457	4	19	1,629	98%	1	1.0%
Ontology (full)	**0.778**	15	19	**1,160**	**70%**	6	**100%**

they quickly identify an initial candidate, lack of complete ontological context causes subsequent iterations to plateau without further design improvement.

Mass efficiency and target attainment. Four of the five conditions produce their best design near the upper boundary of the mass budget (94% to 98% of the cap), adding material to attain structural stiffness. In contrast, the full Ontology condition achieves its maximum performance at 1,160 kg, representing 70% of the cap and 91% of the baseline mass, making it the only condition to improve performance objectives while simultaneously reducing material. Evaluated against absolute target thresholds rather than relative rankings, no condition satisfies all three constraints simultaneously. The full Ontology condition meets both the stress (241.4 MPa) and eigenvalue (336,500) targets while missing the frequency constraint (4.121 Hz against 7.5 Hz). The Baseline condition satisfies only the stress target (232.6 MPa), missing both eigenvalue (158,900 vs. 162,000) and frequency (3.159 Hz). The remaining three intermediate conditions fail all three targets as shown in Table 15.

Score retention and boundary behavior. Four of the five conditions conclude their runs holding only 0.2% to 1.0% of their peak score, whereas the full Ontology condition retains 100%. In all four failing runs, this performance drop followed a drastic reduction of `BaseWidth` to the minimum bound of 500 (from a baseline near 1,500) applied at iterations 10, 13, 4, and 8, respectively. This parameter change reduced the part roughly three times and increased the maximum stress from 232 MPa to 492 MPa. Consequently, different strategy exploration by the full ontology condition led to retention of the best score.

Computational and token cost. Input token usage scales directly with each condition, rising from 691,271 for Baseline and 540,150 for Information to 1,119,775, 1,231,466, and 1,619,507 across the three Ontology variants. The full Ontology condition consumes 2.34$\times$ the input tokens of the Baseline while issuing 85 ontology queries. However, the frequency of tool calls remains nearly invariant across all conditions (ranging between 64 and 71 calls), indicating that the additional computational overhead stems entirely from expanded context length rather than increased solver invocation.

I Design Tasks with GPT-5.6 Sol

To test how the design results depend on the model, the design tasks were repeated with GPT-5.6 Sol, the highest-scoring model on the interpretation benchmark (Appendix C). Each task was run with five seeds per condition, using the same harness, parts, and requirements as the Opus 4.6 runs. Table 16 reports the results.

Limited material removal. On the helicopter pitch arm, every GPT-5.6 Sol run satisfied the stress and displacement limits, but no run removed more than 0.4% of the starting mass. Opus 4.6 removed 3.72% as Baseline and 9.32% as Ontology, and 3.24% on average even when capped at 10 iterations (Appendix E). The GPT-5.6 Sol result is also nearly identical across all three conditions, with mean mass changes between -0.34% and -0.38%. This suggests that its outcome on this task is therefore set by the model's own design approach, which stops short of removing material, rather than by the reasoning structure it is given.

Reaching targets by adding material. Where mass is not constrained, GPT-5.6 Sol meets its targets by adding material. On the coupled variant, its final designs are 2.5 to 22 times the starting

Table 16: Design tasks with GPT-5.6 Sol, five seeds per condition, reported as mean ± standard deviation. Iters is the number of iterations before the agent stopped. On the vibration task every run reached the 7.5 Hz target, so mass is the outcome. The coupled task is the variant described in the text.

		Baseline		Ontology	
Task	Metric	Outcome	Iters	Outcome	Iters
Helicopter pitch arm	Mass $\Delta\%$	-0.34 ± 0.06	28.6 ± 10.4	-0.35 ± 0.06	27.4 ± 9.2
Buckling	Eigenvalue $\Delta\%$	$+625 \pm 715$	38.8 ± 9.4	$+206 \pm 123$	22.8 ± 4.7
Vibration	Mass at 7.5 Hz (kg)	$2{,}539 \pm 1{,}530$	18.6 ± 4.0	$3{,}151 \pm 1{,}117$	13.6 ± 1.1
Fatigue	Peak stress (MPa)	253 ± 146	22.0 ± 4.8	212 ± 181	18.0 ± 4.9
Coupled (variant)	Combined score	0.50 ± 0.12	30.0 ± 8.5	0.40 ± 0.20	24.6 ± 9.0
	Final mass (kg)	$14{,}763 \pm 8{,}137$		$10{,}887 \pm 6{,}331$	

mass, about twice the mass Opus 4.6 added on the same variant (4.1 and 5.1 times). On the fatigue task, the runs that reached the endurance limit, and with it infinite life, did so at 27 to 54 times the starting mass. These designs satisfy the stress or stiffness targets through bulk rather than through the redistribution of material that the pitch arm explored.

Structured inference shortens the search. Within GPT-5.6 Sol, the Ontology condition reached its stopping decision in fewer iterations than Baseline on all five tasks (stratified permutation test, $p = 0.003$) and with fewer tool calls ($p = 0.010$), with no significant difference in outcome on any task. On the buckling task the reduction is from 38.8 to 22.8 iterations ($p = 0.012$). The Information condition, which carries the same five-step inference chain, shortens the search by a similar amount ($p = 0.003$), so the saving comes from structuring each design change as an inference chain.

Interpretation and action are separate capabilities. GPT-5.6 Sol interprets simulation results more accurately than Opus 4.6 (63.9% against 54.6% on the benchmark), yet removes a tenth as much mass from the pitch arm. The model that reads simulation results best is not the one that designs most efficiently. This is consistent with evaluating observation and action separately, and with the premise of this work that design quality depends on the reasoning that connects an observation to a design change, not on the observation alone.

J Taxonomy Usage and Inference Structure

Table 17 reports how the three conditions with ontology access used it on the multi-objective coupled T-bracket task. Baseline and Information are omitted because neither is given the vocabulary and neither emits any classification. Each of the three runs made 19 design proposals, and each proposal carries a classification payload naming one category per node type.

What the columns measure. *Queries issued* is the number of taxonomy tool calls made during the run, and *last query at* is the final iteration in which any such call was made, out of 19. *Chain completeness* is the fraction of the 19 proposals that carry a category for all five node types. Each run therefore has 95 chain slots, being 19 proposals across 5 node types, and *vocabulary fidelity* is the fraction of those 95 slots holding an exact entry from the 157-category vocabulary. *"Other" used* counts slots filled with the *Other (Specify)* fallback. *Distinct categories* counts the vocabulary entries committed to at least once.

Vocabulary use and inference-chain quality. Table 17 reports how the three conditions with vocabulary access used it. Chain completeness is the fraction of design proposals carrying a category for all five node types. Vocabulary fidelity is the fraction of the 95 available chain slots, being 19 proposals across 5 node types, filled with an exact category from the controlled vocabulary rather than an invented label. Classifications are recovered from the payload attached to each design proposal; the Baseline and Information conditions emit none, as expected, since neither is given the vocabulary.

Table 17: Taxonomy usage and inference-chain quality on the coupled T-bracket task, one run per condition. Only the three conditions with vocabulary access are shown. Distinct categories counts the vocabulary entries committed to at least once, out of 157.

Condition	Queries issued	Last query at iter.	Chain completeness	Vocabulary fidelity	"Other" used	Distinct categories
Ontology (vocabulary)	46	18	89%	62%	8	14
Ontology (coupled)	70	**10**	68%	41%	0	8
Ontology (full)	85	17	**100%**	**82%**	**0**	9

Gap between quality metrics indicates label robustness. Because completeness and fidelity are computed over the same 95 schema slots, the difference between them represents the proportion of slots populated with labels outside the predefined vocabulary. This gap spans 27 percentage points for Ontology (vocabulary), 27 points for Ontology (coupled), and 18 points for Ontology (full). Thus, every condition generates out-of-vocabulary labels, with the full Ontology condition exhibiting the lowest rate. Furthermore, the "Other" column reveals that only a single condition utilized the sanctioned fallback category for unlisted items; a zero in this column therefore indicates unconstrained label creation rather than perfect coverage by the standard vocabulary. This indicates that the model was able to identify additional nodes that did not perfectly align and map them to existing nodes in the ontology suggesting robustness of the structured inference.

Increased query volume does not imply superior ontology utilization. Although total ontology queries increase across variants (rising from 46 (vocabulary) to 70 (coupled) and 85 (full)) chain quality does not scale monotonically. The coupled variant issues 52% more queries than the vocabulary variant yet scores lower across both metrics: 68% vs. 89% on completeness and 41% vs. 62% on fidelity. Instead, chain quality correlates strongly with the temporal persistence of ontology interaction. The two conditions that maintain querying through iterations 17 and 18 achieve 100% and 89% completeness, whereas the coupled variant) which ceases querying at iteration 10 (reaches only 68%. This suggests its lower performance stems from mid-run disuse of the ontology rather than sustained, imprecise usage.

Vocabulary use is narrow. The three conditions committed to 14, 8 and 9 distinct categories, which is 5 to 9% of the 157 available. Additional information from the ontology does not broaden this: Ontology (full) issued the most queries, 85, and committed to 9 categories, against 14 for Ontology (vocabulary) on 46 queries. Indicating that exploration of the ontology is conditioned on the specific task (this work explored 3 of the 8 failure modes in the ontology).

The vocabulary fails at the observation step. Off-vocabulary labels are non-uniformly distributed across the design chain. Pooling all three conditions ($N = 49$ filled slots per node type), Figure 7 plots the proportion of slots carrying labels absent from the predefined vocabulary. This out-of-vocabulary rate decreases monotonically along the chain, dropping from 61.2% at OBSERVE to 16.3% at ACT. Although the agent reliably classifies its intended mechanisms and planned actions, it struggles to categorize its perceptual state. Crucially, OBSERVE is the smallest category set in the vocabulary (11 entries, compared to 47 for DESIGN and 39 for ACT), meaning the step providing the fewest predefined options is the most frequently bypassed. Rather than generating arbitrary strings, the created OBSERVE labels are domain-specific: *Performance metric below target*, *Multi-objective plateau*, and *Multi-objective regression* constitute 20 of the 30 instances, each describing a multi-objective optimization state unrepresented in the ontology.

K PROCESS AND REASONING-TRACE MEASURES

Two distinct metric families evaluate the coupled T-bracket task: process measures and reasoning-trace measures. Process measures characterize how each run allocated its mass budget and traversed the multi-objective design space. In contrast, reasoning-trace measures analyze the free-text rationales generated by the agent alongside each design modification. Both metric sets are computed across individual, 19- to 20-iteration runs per experimental condition.

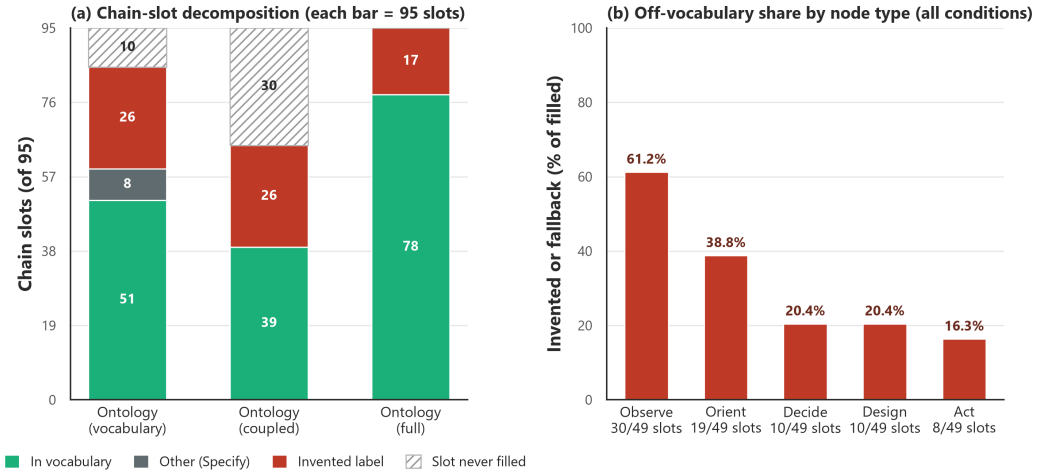

Figure 7: Ontology fidelity on the coupled T-bracket, by vocabulary condition. (a) Each run's 95 chain slots (19 proposals × 5 node types) split, into in-vocabulary, Other (Specify) fallback, invented, and never-filled. (b) Off-vocabulary share by node type: the vocabulary fails most at Observe (61%, only 11 categories) and least at Act (16%). Ontology (full).

Search behavior across the scaffolding ladder. Table 18 presents the process-level metrics, two of which differentiate the experimental conditions most distinctly. The first is the iteration at which a run initially reaches 90% of the mass cap; the threshold beyond which the constraint binds, requiring performance gains to come from material redistribution rather than bulk addition. The second is the feasible-set hypervolume, defined below. Notably, the full Ontology condition is the sole variant that never reaches 90% of the mass cap, yet it achieves the broadest coverage of the multi-objective space.

Table 18: Search behavior on the coupled T-bracket task, one run per condition. Peak cap use is the maximum mass reached as a fraction of the 1,665.39 kg cap. Parameters per proposal is the mean number of parameters altered in a single design change, a measure of how localized each edit was.

Condition	First iter. at ≥90% cap	Peak cap use	Feasible-set hypervolume	Reset requests	Params. per proposal
Baseline	5	95%	0.046	6	2.85
Information	4	96%	0.198	5	2.47
Ontology (vocabulary)	2	107%†	0.001	9	2.71
Ontology (coupled)	4	98%	0.003	7	3.21
Ontology (full)	**never**	**77%**	**0.260**	**2**	**2.05**

† This condition briefly exceeded the mass cap, producing an infeasible design that was not eligible to be the run's best.

Feasible-set hypervolume. Each objective is normalized to $[0, 1]$ as the fraction of the baseline-to-target gap achieved, clamped at both ends, so a value of 1 denotes the target and 0 denotes the starting part. For a run, let S be the set of normalized objective vectors from its feasible iterations. The feasible-set hypervolume is the volume of the region of the unit cube dominated by S, taken with respect to the origin:

$$\mathrm{HV}(S) \;=\; \mathrm{vol}\left(\bigcup_{s \in S} [0, s_1] \times [0, s_2] \times [0, s_3] \right). \tag{1}$$

This metric is independent of the weight vector used in the composite score, evaluating the entire trajectory of generated designs rather than a single optimum. Consequently, it distinguishes a run that discovers a solitary high-performing design from one that systematically maps a broad trade-off

frontier; a distinction captured by composite scores alone. Hypervolume values are computed on a discretized grid 96^3, which yields a spatial resolution finer than 10^{-3}, well below the magnitude of the reported performance differences.

Reset requests and edit locality. Reset requests quantify invocations of the tool that restores the baseline (iteration-0) feature set. These are extracted directly from the model's tool-call logs rather than solver logs, as a model reset does not execute a numerical solve. Edit locality is measured by the mean number of parameters altered per design modification. The full Ontology condition exhibits the lowest values across both metrics, executing only 2 reset requests and modifying just 2.05 parameters per candidate design, while uniquely remaining below the mass cap throughout its run. The two metrics do not otherwise co-exist: the Ontology (vocabulary) variant issues the highest number of reset requests (9) while ranking third in parameters per design modification. Given the five-point sample size, no statistical association can be established between these behaviors.

Reasoning traces. Across all five conditions, the runs produced 96 candidate designs, each accompanied by a free-text rationale. The metrics detailed below are evaluated over these 96 rationales to quantify what the agent declared in its reasoning, independently of physical design performance.

Intent-action alignment. Each rationale is parsed to extract the agent's declared actions, which are then compared against the specific parameters and features modified in the corresponding design edit. Parameter targets are matched by name. *Did what it said* measures the proportion of declared target modifications that were actually executed. *Said what it did* measures the proportion of executed modifications that were declared prior to execution. Finally, *Direction agreed* measures the proportion of matched parameters (whose direction of change is determinable) for which the declared direction matches the executed edit.

Table 19: Intent-action alignment on the coupled T-bracket task, computed over 96 proposal rationales. The Baseline row is not directly comparable with the other four; see the note below.

Condition	Did what it said	Said what it did	Direction agreed
Baseline[‡]	0.86	0.79	**0.96**
Information	0.83	0.93	0.75
Ontology (vocabulary)	0.80	0.95	0.93
Ontology (coupled)	0.71	**0.97**	0.81
Ontology (full)	**0.90**	0.90	0.92

[‡] The four structured conditions state their intended action after an explicit marker, and the span read from each rationale is 48 to 51% of its text. Baseline emits no such marker and is read through a sentence filter that retains 76%, a larger and differently selected span.

Because extraction mechanics vary by condition, the Baseline row should not be ranked directly alongside the structured conditions. Among the four structured variants, the full Ontology condition states its intended actions and subsequently executes them most consistently; achieving a score of 0.90, compared to 0.71-0.83 for the remaining variants. Conversely, the coupled Ontology condition is the least likely to state an action prior to execution.

L Qualitative Analysis of Reasoning Traces

This section examines the agents' written rationales across experimental campaigns rather than their quantitative outcome metrics. Quotations are drawn verbatim from execution logs. Because each condition represents a single trajectory, these traces illustrate qualitative mechanisms rather than precise statistical rates; where supported by aggregate counts across runs, those totals are provided.

Reaching for a new lever at a constraint boundary. On the helicopter pitch arm task, optimization eventually encounters a state where mass sits on the displacement limit and no additional material can be removed using the active strategy. Conditions respond to this boundary differently. Baseline treats the boundary as terminal: *"Optimized to the practical limit of the displacement constraint. Further mass reduction is not achievable without violating the constraints."* Information

arrives at the same conclusion with explicit reasoning, inferring that any pocket configuration yielding a mass below ≈ 1.262 kg will violate the limit, and reverts to its best feasible design. Conversely, Ontology treats the constraint as a coupled state governed by other parameters, enlarging fillet radii to increase local stiffness, converting the resulting margin into displacement headroom, and consuming that headroom via a deeper pocket. This distinction, between accepting a binding constraint and actively searching for parameters that relax it, underpins the performance gap on this task.

Locating the governing failure mode. On the buckling task, all three conditions eventually identify the T-flange rather than the base as the critical buckling element, but they differ sharply in iteration latency. Baseline identifies the mechanism at iteration 20, executing a 1 mm adjustment to rib height. Information identifies it at iteration 29 alongside a column-buckling governing equation. Ontology isolates the failure mode at iteration 2 by inspecting the modal separation between the first two eigenvalues (52,960 vs. 481,100), concluding that a gap of this magnitude indicates a localized web mode rather than a global structural mode, making base modifications ineffective. Evaluating the full eigenvalue spectrum rather than a single primary eigenvalue is the critical diagnostic step, an approach unique to the Ontology condition on this task.

Budget headroom precedes divergence on the coupled task. On the coupled T-bracket task, all four conditions that ultimately lose their best design reach the mass cap early in the run. The first iteration at or above 90% of the mass cap occurs at iteration 2 for Ontology (vocabulary), iteration 4 for Information and Ontology (coupled), and iteration 5 for Baseline; by contrast, Ontology (full) never exceeds 77% of the cap. Once the mass budget is exhausted, additive material allocation is precluded, restricting optimization to base-plane adjustments. Ontology (coupled) states this explicitly at iteration 5, noting that near-total budget consumption limits traditional material addition. Ontology (full) avoids this regime entirely. This causal sequence, from mass efficiency to budget headroom to avoiding forced base modifications, is clearly legible in both the reasoning traces and Table 18.

Identical coupling entries drive opposing decisions. Ontology (coupled) and Ontology (full) both accessed the coupling lookup tool regarding whether to increase base thickness or add a stiffening rib. The resulting feedback altered agent behavior in opposite directions, initiating divergent optimization trajectories.

At iteration 11, Ontology (full) increased base thickness from 45 mm to 75 mm and evaluated the result. Querying the coupling entry for thickness increases yielded a side-effect warning: stiffer sections attract higher local loads and elevate stress at adjacent transitions. The agent logged:

> OBSERVE: ... Going from BT=45→75 worsened stress from 270.2→294.7 while only improving frequency from 3.64→3.78, very poor tradeoff. DESIGN: Stress redistribution feedback, the thicker base attracted more load, increasing peak stress. ... We need strategies that raise I without proportional mass increase. *Ontology (full), iteration 11*

The agent reverted base thickness to 45 mm, added a rib, and appended additional ribs over the following two iterations, reaching its peak score at iteration 15.

At iteration 4, Ontology (coupled) added a single rib and observed a peak stress increase from 264 MPa to 287 MPa. Querying the coupling entry for stiffeners returned a warning that rib terminations create stress concentration sites vulnerable to fatigue. The agent logged:

> This confirms the rib created stress concentrations at its runouts. Let me remove the rib and try a different strategy. *Ontology (coupled), iteration 4*

The agent deleted the rib and never accumulated more than one rib for the remainder of the run. In both instances, the tool was executed competently. In one trajectory, it corrected a strategy that empirical solver feedback proved inefficient; in the other, it provided a confirmatory explanation for a single noisy observation, licensing the premature abandonment of a structural feature that drove peak performance in the opposing run. Causal frameworks highlighting side effects increase both behaviors, offering a valid rationale to abandon a strategy as readily as a rationale to pursue it.

Structural scaffolding with and without behavioral constraint. All three vocabulary conditions emit all five labeled steps in every classified candidate design, demonstrating that emission alone is not a differentiating factor. The functional distinction lies in whether intermediate chain reasoning responds to top-level observations. In Ontology (vocabulary), the Orient label reads *Insufficient frequency margin* across its first ten classified candidate designs, including iteration 10 where the Observe label records *Solver failure/non-convergence*. A solver error is thus misdiagnosed as a frequency margin deficit, carrying forward a static diagnosis rather than deriving one from solver state, which sever the link between observation and subsequent action. In Ontology (full), the Orient label dynamically tracks the Observe label, shifting to solver and geometry categories during failure sequences and to *Stress redistribution feedback* at iteration 11; the precise step where diagnosis reverses the design action.

Recovery dynamics. Reset requests restore the iteration-0 feature set, stripping all accumulated geometric features. Ontology (full) experienced the longest continuous sequence of solver failures across all conditions yet issued only two resets, successfully recovering via a single-parameter adjustment before achieving its peak design. In contrast, Ontology (vocabulary) issued nine resets across iterations 7-19, repeatedly deleting the mass-efficient features its own rationales identified as necessary. Cumulative reset counts across the ladder were 2, 5, 6, 7, and 9 for Ontology (full), Information, Baseline, Ontology (coupled), and Ontology (vocabulary), respectively.

Ontological gaps during non-design recovery regimes. A substantial fraction of iterations across all conditions was spent recovering from solver failures or invalid parameter states rather than performing constructive design synthesis. The taxonomy lacks explicit Design categories for restoring feasibility or reverting to known-good states, Orient categories for solver failures, and Observe categories for metric targets being unmet (its 11 Observe entries pertain exclusively to analytical findings or test results). The reasoning traces document the agent discovering these limitations. Ontology (vocabulary) executed 34 vocabulary searches, 14 of which queried recovery terminology such as *restore*, *revert*, *proven*, *solver failure*, *convergence*, and *feasibility*. Eight of these queries occurred in iteration 11, where all five schema slots defaulted to *Other (Specify)*. No other condition searched for recovery terms. All three structured conditions subsequently generated custom out-of-vocabulary labels to bridge these gaps, yielding off-vocabulary rates of 27%, 27%, and 18% across chain slots (Figure 7). This reflects a structural limitation of the classification instrument itself, indicating that future campaigns require explicit recovery and meta-design branches within the ontology.

Scope and generalizability. Each condition comprises a single trajectory per task, confounding condition and individual run variance. These qualitative traces establish that specific behavioral mechanisms occurred and are legible within execution records, but they do not establish empirical frequency under repeated sampling.