# CADEngBench: Can AI Systems Co-Author Engineering Designs? A Hierarchical Benchmark for Physics-Verified Parametric CAD Generation

**Anonymous Authors**[1]

## Abstract

AI systems are increasingly proposed as co-authors for engineering design, yet current CAD evaluations largely reward executable geometry rather than reusable, physics-valid parametric models. We introduce CADEngBench, a hierarchical benchmark of 40 mechanically meaningful CadQuery tasks spanning 12 engineering domains and 17 physics tags. CADEngBench evaluates generated designs across four layers: $L_0$ compilation and topological validity, $L_1$ geometric fidelity, $L_2$ parametric integrity via perturbation robustness and Z3 SMT constraint verification, and $L_3$ closed-form physics verification. Across five representative models evaluated with a standardized prompting and scoring pipeline, Claude Sonnet 4.6 achieves the strongest end-to-end performance at $22/40$ passes ($55.0\%$). The benchmark's most diagnostic result is a specialist-generalist inversion: CADCoder attains $39/40$ at $L_0$ but only $2/40$ at $L_2$ and $1/40$ overall, demonstrating that runnable CAD code is not equivalent to engineering-grade CAD. Prompt underspecification is catastrophic, with $1/60$ overall passes at `level_1` versus $20/60$ at `level_3`, and repeated sampling improves frontier hosted models but not CADCoder. These findings identify $L_2$ parametric integrity as the principal failure boundary for current systems and position CADEngBench as a benchmark for whether AI systems can co-author reusable, physics-validated engineering designs.

## 1. Introduction

Programmatic CAD generation is an attractive AI-for-science target because it converts natural-language design intent into editable engineering artifacts. Existing evaluations, however, often reward the wrong minimum condition: executable or visually plausible geometry. For engineering design, a CAD program is useful only if it also preserves specified dimensions, propagates parameter changes through the intended constraints, and satisfies the physical assumptions attached to the object. A model that emits a runnable script but hard-codes dimensions, breaks under parameter edits, or violates a simple stress or flow criterion has not co-authored an engineering design; it has produced disposable geometry.

We introduce CADEngBench, a benchmark for this stronger notion of CAD co-authorship. The benchmark contains 40 CadQuery generation tasks spanning 12 engineering domains, 17 physics tags, and three difficulty tiers. Each task is packaged as a synchronized object contract: natural-language prompt variants, semantic metadata, machine-readable constraints, and an executable ground-truth CadQuery implementation. CADEngBench then evaluates generated programs with a four-layer hierarchy: $L_0$ compilation and topological validity, $L_1$ geometric fidelity, $L_2$ parametric integrity through perturbation robustness and Z3 SMT constraint checks, and $L_3$ closed-form physics verification. Overall success is the conjunction over the four layer predicates:

$$P_{L_i}(x) = \mathbf{1}\{x \text{ passes layer } L_i\}, \quad i \in \{0,1,2,3\},$$

$$P_{\text{overall}}(x) = P_{L_0}(x) \wedge P_{L_1}(x) \wedge P_{L_2}(x) \wedge P_{L_3}(x).$$

The hierarchy is deliberately not a single monotone geometry score. $L_2$ and $L_3$ test different engineering properties: a nominal artifact may satisfy a closed-form structural check while still failing as reusable CAD because its constraints do not survive edits. This separation is the central methodological choice in CADEngBench and the main reason the benchmark exposes failures hidden by code-validity-only or geometry-only evaluation.

We evaluate GPT-5.1, Claude Sonnet 4.6, Gemini 2.5 Pro, Qwen3 Coder 30B A3B, and CADCoder under a standardized `level_3` prompting and scoring pipeline. Claude Sonnet 4.6 leads the full benchmark with $22/40$ overall passes, but the more diagnostic finding is a specialist-generalist inversion: CADCoder reaches $39/40$ at $L_0$ yet only $2/40$ at

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Submitted to the AI for Science workshop (ICML 2026).

$L_2$ and $1/40$ overall. If evaluation stopped at executability, CADCoder would appear dominant; under engineering evaluation, it is last. Ablations reinforce the same message: underspecified prompts collapse to $1/60$ overall passes, while repeated sampling improves frontier hosted models but does not repair CADCoder's parametric failure mode.

Our contributions are:

- A 40-object, physics-tagged CadQuery benchmark with prompt, metadata, constraint, and ground-truth artifacts for each object.
- A hierarchical evaluator separating compilation, geometry, parametric integrity, and closed-form physics verification.
- A five-model study showing that $L_2$ parametric integrity is the dominant boundary between runnable CAD and engineering-grade CAD.
- Prompt-specificity and multigeneration ablations, with preserved raw generations, extracted scripts, manifests, evaluator JSON, and aggregate tables for auditability.

## 2. Related Work

**Benchmarks for generated programs and agents.** HumanEval introduced pass@k-style functional evaluation for generated code (Chen et al., 2021), and LiveCodeBench emphasizes contamination-aware, multi-capability code evaluation (Jain et al., 2024). HELM argues for scenario- and metric-decomposed evaluation rather than single-score leaderboards (Liang et al., 2023), while SWE-bench and MLE-bench show the value of realistic, end-to-end task environments (Jimenez et al., 2024; Chan et al.). CADEngBench adopts these benchmark lessons, but applies them to CAD-specific engineering properties: dimensions, parametric editability, and physics validity. The missing piece in these benchmark traditions is that CAD correctness is not only functional input-output behavior; it is also whether a generated design program preserves intended engineering relations when edited.

**Text-to-CAD generation.** The recent LLM-for-CAD survey of Zhang et al. (2026) organizes the area around applications including CAD code generation, parametric CAD generation, model evaluation, data generation, image generation, and text generation. Within this landscape, CAD-Coder is the closest published specialist baseline to CADEngBench because it targets CadQuery code generation with chain-of-thought and geometric reward modeling (Guan et al., 2025). These works establish LLM-based CAD generation as an active research area, but their reported metrics still primarily emphasize command or parameter accuracy, renderability, sequence validity, visual similarity, or geometric reward. CADEngBench targets the evaluation gap left by those metrics: whether generated CAD programs preserve prompt-specified engineering constraints under perturbation and satisfy object-aware physics checks.

**Formal and physics-aware verification.** Our evaluator uses Z3 SMT checks to test symbolic constraint satisfaction, following the broader tradition of solver-backed program and property verification (de Moura & Bjørner, 2008). CADEngBench does not treat solver checks as a standalone proof system for arbitrary CAD; they are one component of a layered engineering evaluator combining executable topology, geometric measurements, parametric perturbations, and closed-form physics surrogates. The verification target is therefore deliberately narrower than full formal CAD correctness: it is to expose whether model-generated scripts encode the specific constraints that make the object reusable.

## 3. CADEngBench

### 3.1. Dataset and Object Contracts

CADEngBench is designed to evaluate engineering CAD generation as program synthesis under constraints. The benchmark contains 40 CadQuery tasks spanning 12 engineering domains, 17 implemented physics tags, and three difficulty tiers. The object set is intentionally audit-sized but not toy-sized: it includes beams, brackets, pressure vessels, columns, foundations, fluid devices, thermal parts, gears, shafts, threads, retaining structures, heat exchangers, arches, and thin-walled torsion members. Thus the benchmark tests whether a model can preserve object-specific engineering relations, not whether it can draw a generic solid.

The central dataset unit is an *object contract*. For object $o$, we store

$$c_o = \left(p_o^{(1)}, p_o^{(2)}, p_o^{(3)}, s_o, q_o, g_o\right),$$

where $p_o^{(1:3)}$ are the prompt variants, $s_o$ is the structured specification, $q_o$ is the constraint contract, and $g_o$ is executable reference CadQuery. The model receives only one prompt $p_o^{(\ell)}$ and must produce a standalone CadQuery script $x$ ending in `result`. The hidden contract $s_o, q_o, g_o$ drives evaluation. This matters because engineering CAD correctness is not exhausted by resemblance to a reference mesh: a correct script must expose the intended parameters, derive dependent dimensions, preserve relational constraints, and remain physically meaningful.

Concretely, each `constraints.json` defines primary parameters, dependent parameters with derivations, relational checks with executable predicates and tolerances, an expected bounding box, and a perturbation specification. A

task is therefore not just "make this shape"; it is a contract over geometry, code structure, and physical behavior:

$$q_o = \{\theta_{\text{primary}}, \theta_{\text{dep}}, r_{1:K}, b, \Delta\theta, \pi_{\text{prop}}\}.$$

Here $r_{1:K}$ are nominal design relations, $b$ is the dimensional envelope, $\Delta\theta$ names perturbable variables, and $\pi_{\text{prop}}$ specifies which dependent quantities must update when a primary parameter changes.

### 3.2. Prompt and Reference Protocol

The prompt ladder separates task underspecification from model capability. `level_1` gives a terse design request, `level_2` adds the main dimensions and modeling intent, and `level_3` gives the full engineering contract: dimensions, design relations, material/load fields, pass condition, and a request to expose primary dimensions in a parameter dictionary when possible. The main benchmark uses `level_3` because real engineering CAD tasks normally require explicit constraints. The prompt ablation then weakens the same objects to determine whether failures are caused by missing specification or by missing CAD/engineering capability.

The reference scripts follow a common pattern: declare primary parameters, derive dependent dimensions, assert feasibility constraints, construct `result`, and expose a nominal geometry that satisfies all evaluator layers. They are used to calibrate extraction, constraints, expected utilization, and exported artifacts, but model outputs are never compared by code similarity. This avoids a common failure in CAD evaluation: rewarding a visually similar but non-parametric artifact.

### 3.3. Hierarchical Evaluator

Let $x$ be a generated script, $G(x)$ the solid returned by `result`, and $\hat{\theta}(G(x))$ the recovered measurements and object features. The evaluator is not a visual-similarity score. It is a sequence of object-aware tests that separate construction, nominal geometry, parametric editability, and physics validity. This separation is essential: a script can compile and even satisfy a nominal stress check while still being unusable as engineering CAD if it hard-codes dimensions or fails under edits.

The layers intentionally share artifacts but answer different questions. $L_0$ produces the executable solid and topology record. $L_1$ converts that solid into an authoritative measurement namespace for the object contract. $L_2$ edits the script and reruns execution/measurement under perturbed parameters. $L_3$ evaluates the nominal solid against the physics tag. Thus $L_2$ and $L_3$ are not redundant: one tests reusable editability, the other tests nominal engineering validity.

**$L_0$ and $L_1$: construction and nominal design fidelity.** $L_0$ is the entry gate: the script must run, define `result`, and produce a CadQuery `Workplane` or shape that OpenCascade can validate as a solid. The evaluator records volume, bounding box, face/edge/vertex counts, Euler-characteristic diagnostics, open edges, and non-manifold indicators. It also captures scalar variables and parameter dictionaries because those are later used by the perturbation evaluator.

$L_1$ evaluates whether the nominal solid encodes the specified design relations. It computes bounding-box and ISO 2768-m diagnostics, detects benign axis permutations, and uses object-specific feature extractors for quantities such as bore spacing, hole position, wall thickness, diameter ratio, fin count, throat radius, or dependent inner radius. The extracted measurements are merged with authoritative prompt quantities into a namespace and checked against the relational predicates $r_k$ from `constraints.json`:

$$P_{L_1}(x) = \mathbf{1}\left\{r_k(\hat{\theta}(G(x))) = 1 \ \forall k\right\}.$$

Code-structure diagnostics such as parameter-dictionary presence, named-variable coverage, formula count, and normalized compression distance are logged for analysis, but they do not replace geometric and relational checks.

**$L_2$: parametric integrity.** $L_2$ asks whether the same relations survive edits. For each perturbable parameter $\theta_j$, the evaluator samples

$$\theta_j' \in [0.75\theta_j, 1.25\theta_j],$$

using a deterministic Morris-design perturbation matrix. It substitutes sampled values into the script through parameter dictionaries, named top-level variables, or a value-matched literal fallback, then re-executes $L_0$ and $L_1$ on each modified script. Infeasible perturbations under the object contract are skipped; substitution, execution, and feature-extraction failures are retained as audit records. $L_2$ reports behavioral constraint propagation and solver-backed constraint preservation:

$$\text{PIS}(x) = \frac{n_{\text{prop}}}{n_{\text{beh}}}, \qquad \text{ZSR}(x) = \frac{n_{\text{proved}}}{n_{\text{checked}}}.$$

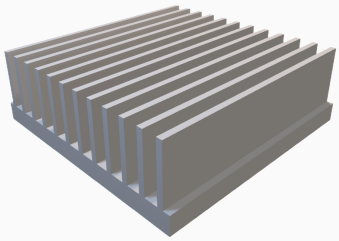
$$S_{L_2}(x) = \begin{cases} \text{PIS}(x), & n_{\text{checked}} = 0, \\ \frac{1}{2}\left(\text{PIS}(x) + \text{ZSR}(x)\right), & n_{\text{checked}} > 0. \end{cases}$$

The behavioral term checks whether expected propagation rules, such as `fin_height = 3 * base_thickness`, remain true across successful perturbation samples. The Z3 term parses generated assignments and assertions into symbolic relations, searches for counterexamples over the perturbation bounds, and counts only formula-backed proofs as preserved constraints. The scalar score determines the layer pass.

*Table 1.* CADEngBench benchmark-suite summary. The suite is organized around engineering contracts, not isolated prompts or meshes.

| Axis | Benchmark contents | Evaluation role |
|---|---|---|
| Task set | 40 CadQuery program-generation objects; 11 Tier 1, 22 Tier 2, 7 Tier 3 | Supports aggregate, tier-wise, and object-level analysis without losing manual auditability |
| Engineering domains | 12 domains; largest groups are mechanical (17), civil/fluid (4 each), automotive/thermal (3 each) | Prevents the benchmark from collapsing into one CAD family or one extraction strategy |
| Physics tags | 17 tags; most common are beam (7), pressure_vessel (7), column (4), fluid (4), bracket (3) | Selects object-specific feature extractors and the $L_3$ closed-form physics verifier |
| Object contract | `prompts.json`, `spec.json`, `constraints.json`, and `ground_truth.py` for every object | Links language, dimensions, relational constraints, perturbation variables, reference CAD, and physics metadata |
| Prompt ladder | Mean prompt lengths are 23, 44, and 104 words for `level_1`, `level_2`, and `level_3` | Separates underspecification effects from model failures in CAD construction or engineering reasoning |
| Preserved artifacts | Raw responses, extracted scripts, execution logs, geometry exports, JSON reports, aggregate tables | Makes failures inspectable and supports benchmark validity audits |

**Example object contract: `finned_heat_sink`** *thermal domain; heat_transfer tag*



Ground-truth render

**Prompt** "Create a finned heat sink with rectangular fins: base width = 60 mm (X-axis); base depth = 60 mm (Y-axis, flow direction); base thickness = 5 mm (Z-axis); fin thickness = 1.5 mm; fin pitch = 5.0 mm (centre-to-centre, must be $\geq$ 2.5 mm); fin height = 3× base thickness = 15 mm; `num_fins = floor(base_width / fin_pitch)` = 12; aluminium 6061 ($k$ = 167 W/mK); under natural convection ($h$ = 25 W/m$^2$K), $\eta = \tanh(mL)/(mL) \geq 0.3$. Fins should be arrayed along X on the top face; use a parameter dictionary."

**Parameters** Perturb `base_width` and `base_thickness`; derive `fin_height` and `num_fins`.

**Relations** Enforce `fin_height = 3 * base_thickness`, `fin_pitch >= 2.5`, and `num_fins = floor(base_width / fin_pitch)`.

**CAD pattern** `params` → derived fin height/count → feasibility assertions → base box with rectangular fin array.

**Evaluation** $L_1$ extracts fin geometry; $L_2$ checks propagation under width/thickness edits; $L_3$ verifies heat-transfer utilization.

*Figure 1.* Example CADEngBench object contract. The render visualizes the ground-truth reference for readability; scoring uses generated CadQuery execution, extracted constraints, parameter perturbations, and physics verification, not image matching.

**$L_3$: physics verification.** $L_3$ dispatches to one of the 17 implemented physics-tag evaluators. The registry covers Euler-Bernoulli beam bending, pressure-vessel hoop stress, bracket von Mises stress, column buckling, Venturi/weir discharge, heat-transfer area, gear or housing bending, thread engagement, retaining-wall overturning, foundation bearing, arch compression, and thin-walled torsion. Each check returns a governing utilization ratio:

$$P_{L_3}(x) = \mathbf{1}\{\mathrm{UR}(G(x)) \leq 1\}.$$

Each $L_3$ report includes extracted geometry, the physics family, the governing equation family, a governing case, and an error code when extraction or verification fails. These checks are closed-form engineering surrogates rather than CFD, which keeps the benchmark deterministic, inspectable, and inexpensive to run across many model generations.

**Overall score and audit trail.** Overall success is deliberately strict:

$$P_{\mathrm{overall}}(x) = P_{L_0}(x) \wedge P_{L_1}(x) \wedge P_{L_2}(x) \wedge P_{L_3}(x).$$

CADEngBench also reports autonomy labels: *co-author* requires $L_3$ pass with PIS $\geq$ 0.7 and ZSR $\geq$ 0.5, *tool* denotes nominal physics validity without parametric-autonomy strength, and *tool-failing* denotes structural failure or non-evaluation. Every run preserves the raw model response, extracted script, execution logs, JSON reports, and aggregate tables, so failures can be audited rather than inferred from a single score.

| Stage | Question answered | Implementation signals | Artifact |
|---|---|---|---|
| Model output | Did the response contain a runnable CAD program? | Extract Python, normalize imports, require `result`, capture parameter dictionaries and globals. | script $x$ |
| $L_0$ runtime | Does $x$ construct a valid solid? | Execute CadQuery/OCC; check solid validity, bbox, topology, volume, open edges, non-manifold diagnostics. | solid $G(x)$ |
| $L_1$ measurement | Does nominal geometry satisfy the object contract? | Apply axis policy and feature extractor; build measurement namespace; check relational predicates $r_{1:K}$. | measured $\hat{\theta}$ |
| $L_2$ editability | Do constraints survive parameter edits? | Morris perturbations; substitute dicts/variables/literals; rerun $L_0$/$L_1$; compute robustness, PIS, and ZSR. | $S_{L_2}$ report |
| $L_3$ physics | Is the nominal artifact physically valid? | Dispatch by physics tag; use extracted geometry in closed-form governing equation; compute utilization ratio. | physics report |
| **Preserved trail:** raw response, extracted script, generated CAD, JSON reports, perturbation records, Z3 outcomes, and physics report. | | | |

*Figure 2.* Implementation-level evaluator dataflow. CADEngBench scores generated programs through executable artifacts and object contracts, not image similarity or code similarity to the reference.

**Algorithm 1** $L_2$ parametric-integrity evaluation.

**Require:** script $x$, contract $q_o$, nominal $L_1$ namespace
detect substitution mode: `dict`, `variable`, or value-match fallback
build Morris samples $\Theta' = \{\theta'_m : \theta'_j \in [0.75\theta_j, 1.25\theta_j]\}$
**for** $\theta'_m \in \Theta'$ **do**
  derive effective namespace using $\pi_{\text{prop}}$
  **if** contract predicates are infeasible **then**
    skip sample and log infeasibility
  **else**
    substitute $\theta'_m$ into $x$ and rerun $L_0$/$L_1$
    store success/failure and recovered namespace
  **end if**
**end for**
PIS $\leftarrow$ fraction of behavioral propagation rules preserved
ZSR $\leftarrow$ fraction of formula-backed constraints proved by Z3 $S_{L_2}$, robustness, perturbation failures, and counterexamples

## 4. Evaluation Protocol

The main condition evaluates one generation per object on all 40 tasks with `level_3` prompts and $L_0$–$L_3$ scoring. We compare GPT-5.1, Claude Sonnet 4.6, Gemini 2.5 Pro, Qwen3 Coder 30B A3B, and CADCoder via their respective hosted API or local-runtime paths. Provider wrappers are limited to formatting/runtime compatibility; all samples use the same code extraction, `result` binding, execution, and scoring pipeline. These axes are chosen to separate three distinct questions: default end-to-end reliability on the full benchmark, sensitivity to engineering specification detail, and latent capability under repeated sampling.

For model $m$ and object set $\mathcal{O}$, layer counts are sums of binary layer predicates:

$$C_i(m) = \sum_{o \in \mathcal{O}} P_{L_i}(x_{m,o}), \quad C_{\text{all}}(m) = \sum_{o \in \mathcal{O}} P_{\text{overall}}(x_{m,o}).$$

We report layer counts, overall pass, co-author count, failure taxonomy, tier/domain/tag slices, and $L_2$ decomposition. For multigeneration, object-level pass@$k$ is

$$\text{pass@}k(m) = \frac{1}{|\mathcal{O}|} \sum_{o \in \mathcal{O}} \mathbf{1}\left\{\exists g \le k : \; P_{\text{overall}}(x_{m,o,g}) = 1\right\}.$$

We use object-level bootstrap intervals for per-model pass rates and paired model differences on overall and $L_2$ pass. The 12-object ablation slice contains four objects per tier.

*Table 2.* Evaluation conditions. All samples use the same extraction and layer-scoring pipeline.

| Condition | Scope |
|---|---|
| Main run | 5 models × 40 objects × 1 generation = 200 samples; `level_3` prompts; $L_0$–$L_3$ scoring |
| Prompt ablation | 5 models × 12 objects × 3 prompt levels = 180 samples; compares `level_1`, `level_2`, `level_3` |
| Multigen ablation | 3 models × 12 objects × 3 generations = 108 samples; reports pass@1 and pass@3 |
| Artifacts | 488 evaluated samples total; raw responses, scripts, manifests, JSON reports, aggregate tables |

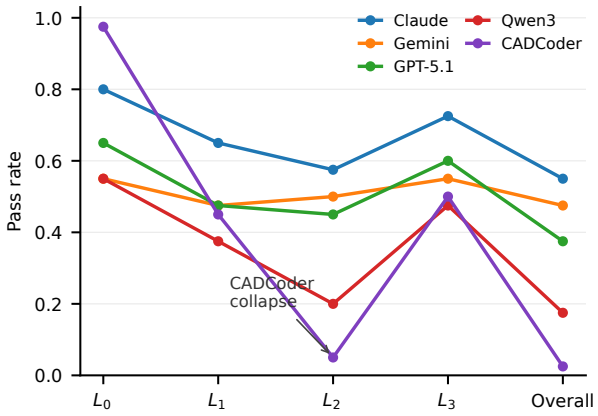



*Figure 3.* Layer profile across the full benchmark.

## 5. Results

### 5.1. The Hierarchy Changes the Ranking

Table 3 and Figure 3 show that CADEngBench is not an executability leaderboard. Claude Sonnet 4.6 is strongest end-to-end with $22/40$ overall passes ($55.0\%$), followed by Gemini 2.5 Pro at $19/40$, GPT-5.1 at $15/40$, Qwen3 Coder at $7/40$, and CADCoder at $1/40$. If the evaluation stopped at $L_0$, the conclusion would invert: CADCoder would rank first with $39/40$ executable/topologically valid scripts.

The hierarchy exposes what executable CAD misses. CADCoder drops from $39/40$ at $L_0$ to $2/40$ at $L_2$ and $1/40$ overall, while the frontier hosted models retain substantially more mass through parametric and physics-aware checks. The autonomy labels reinforce the same point: Claude produces 15 co-author outputs, whereas CADCoder produces none despite 20 tool-level nominal outputs.

The ranking should still be read statistically. Claude's overall margin over GPT-5.1 is 17.5 percentage points with a 95% paired object-level bootstrap interval of $[2.5, 32.5]$, excluding zero. Claude's 7.5 point margin over Gemini has interval $[-12.5, 27.5]$, so the evidence supports Claude as strongest in this run, but not cleanly separated from Gemini.

### 5.2. $L_2$ Is the Main Engineering Bottleneck

*Table 4.* $L_2$ decomposition. PIS measures parametric integrity, ZSR measures Z3 constraint satisfaction, and robustness measures successful sampled perturbation executions.

| Model | $S_{L_2}$ | PIS | ZSR | Robust. |
|---|---|---|---|---|
| Claude Sonnet 4.6 | 0.521 | 0.475 | 0.567 | 0.659 |
| Gemini 2.5 Pro | 0.372 | 0.402 | 0.342 | 0.860 |
| GPT-5.1 | 0.382 | 0.412 | 0.352 | 0.798 |
| Qwen3 Coder | 0.203 | 0.253 | 0.152 | 0.629 |
| CADCoder | 0.074 | 0.025 | 0.123 | 0.026 |

Table 4 explains why $L_2$ tracks end-to-end success. Overall pass counts are close to $L_2$ counts for every model, because a design that loses parametric consistency rarely recovers at the conjunction over all layers. Claude has the highest mean $L_2$ score ($0.521$). GPT-5.1 and Gemini have similar $L_2$ pass counts but different profiles: Gemini has the strongest sampled robustness ($0.860$), while GPT-5.1 is slightly stronger on PIS and ZSR. This suggests that some Gemini scripts continue executing under perturbation without preserving the intended symbolic relations as reliably.

CADCoder is again the diagnostic case. Its mean $L_2$ score is $0.074$, with near-zero PIS ($0.025$) and sampled robustness ($0.026$). The failure is not primarily syntax or CadQuery familiarity; it is the absence of constraint-preserving parametric structure. This also clarifies why $L_3$ can exceed $L_2$: a nominal solid may pass a closed-form physics check while still failing as reusable CAD under parameter edits.

### 5.3. Failure Modes Are Model-Specific

*Table 5.* Grouped first-failure taxonomy on the main 40-object run. Pass denotes overall pass. Build includes API, syntax/runtime, topology, kernel, timeout, and empty-geometry failures; Spec includes dimensional, geometric, and relational failures.

| Model | Pass | Build | Spec | Param. | Physics |
|---|---|---|---|---|---|
| Claude | 22 | 8 | 6 | 4 | 0 |
| Gemini | 19 | 18 | 3 | 0 | 0 |
| GPT-5.1 | 15 | 14 | 7 | 4 | 0 |
| Qwen3 | 7 | 18 | 7 | 7 | 1 |
| CADCoder | 1 | 1 | 21 | 16 | 1 |

Table 5 shows that equal overall failure labels hide different research problems. Gemini and Qwen lose many samples to early construction failures, especially CadQuery/API and topology issues. CADCoder has almost no build failures, but concentrates failures in specification and parametric categories. GPT-5.1 sits between these regimes.

This taxonomy is useful because the fixes are different. Build failures call for better CadQuery grounding and geometric construction planning; specification failures call for stronger dimension and relation extraction; parametric failures call for constraint-preserving code structure. Collapsing these into one failure count would hide which capability is missing. This supports the central benchmark claim: improving CAD generation requires separating construction skill from geometric fidelity, editability, and physics validity.

### 5.4. CADEngBench Is Not Saturated

The full-40 benchmark still has substantial headroom. Twelve objects are not solved by any evaluated model, while nine objects are solved by exactly one model. At the other

*Table 3.* Full 40-object results. Layer columns are pass counts; overall requires all four layers. Co-author/tool/tool-failing are autonomy labels defined by $L_3$ validity and $L_2$ strength.

| Model | $L_0$ | $L_1$ | $L_2$ | $L_3$ | Overall | Co-author | Tool | Tool-failing |
|---|---|---|---|---|---|---|---|---|
| Claude Sonnet 4.6 | 32 | 26 | 23 | 29 | 22 | 15 | 14 | 11 |
| Gemini 2.5 Pro | 22 | 19 | 20 | 22 | 19 | 10 | 12 | 18 |
| GPT-5.1 | 26 | 19 | 18 | 24 | 15 | 12 | 12 | 16 |
| Qwen3 Coder 30B A3B | 22 | 15 | 8 | 19 | 7 | 2 | 17 | 21 |
| CADCoder | 39 | 18 | 2 | 20 | 1 | 0 | 20 | 20 |

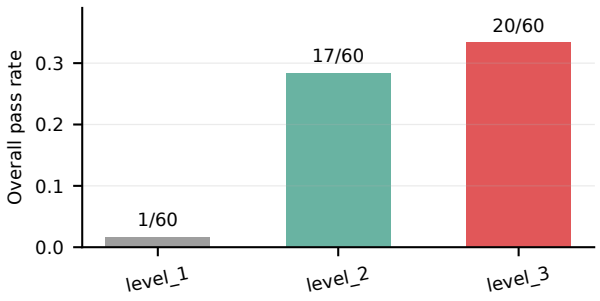



*Figure 4.* Prompt ablation on the 12-object slice. Underspecified prompts nearly eliminate overall success.

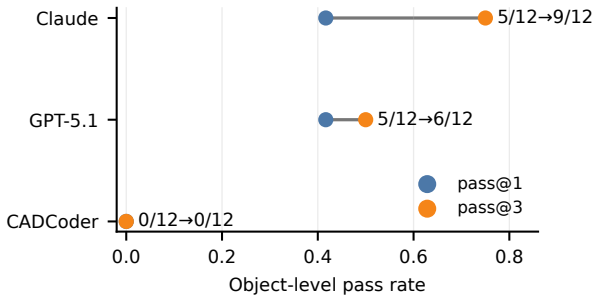



*Figure 5.* Repeated-sampling ablation on the 12-object slice. Sampling improves frontier hosted models but does not rescue CADCoder.

extreme, no object is solved by all five models. This rules out two unhelpful regimes: the benchmark is neither already saturated by frontier systems nor dominated by a small set of universally solved tasks.

The remaining successes are also not redundant across models. Claude contributes six unique object solves and Gemini contributes three, while GPT-5.1 remains competitive in aggregate without contributing a unique solve in this run. This matters for benchmark interpretation: overall pass rate measures average reliability, whereas unique solves measure marginal benchmark coverage.

Difficulty tiers provide a second check on saturation. Claude remains comparatively stable across Tiers 1, 2, and 3 (63.6%, 50.0%, 57.1%), while GPT-5.1 drops from 63.6% on Tier 1 to 14.3% on Tier 3. CADCoder scores 0.0% overall on both Tier 1 and Tier 3 despite its near-perfect $L_0$ rate. The benchmark therefore contains both solvable engineering tasks and hard cases where executability fails to translate into reusable, physics-verified CAD.

### 5.5. Prompt Specificity and Sampling Matter

Figure 4 shows that prompt underspecification is not a usable engineering interface. `level_1` prompts produce only $1/60$ overall passes across models, compared with $17/60$ for `level_2` and $20/60$ for `level_3`. This gap is not simply verbosity sensitivity: `level_3` supplies dimensions, derived relations, loads, and material fields that are required by the evaluator. In this setting, a prompt is part of the engineering specification, not just a natural-language hint.

Figure 5 shows a different effect: repeated sampling changes the ceiling for frontier hosted models. Claude improves from $5/12$ object-level pass@1 to $9/12$ pass@3, and GPT-5.1 improves from $5/12$ to $6/12$. CADCoder remains at $0/12$, indicating that its failure is not low-sample stochasticity but missing reusable parametric engineering structure. Together, the two ablations separate specification failure from generation stochasticity: richer prompts make the task well-posed, while sampling only helps models that already have latent parametric-design capability.

## 6. Qualitative Analysis

**Co-author success requires more than plausible geometry.** Claude's `worm_gear_housing` is a Tier 3 co-author success with $L_2$ score 1.0, PIS 1.0, ZSR 1.0, robustness 0.9167, and $L_3$ utilization ratio 0.384. GPT-5.1's `hollow_rectangular_pier` similarly reaches co-author status with $L_2$ score 1.0, robustness 1.0, and utilization ratio 0.5573. These are not just rendered solids; they preserve editable constraints and satisfy object-specific physics checks.

**Nominal physics validity is not co-authorship.** CADCoder's `thick_walled_cylinder` executes and passes the nominal $L_3$ pressure-vessel check with utilization ratio 0.1169, but fails $L_2$ with score 0.25, PIS 0.0, and robustness 0.0. The generated script is essentially a fixed scaled

cylinder. It is therefore labeled *tool*, not *co-author*: nominally useful, but not a reusable parametric design object.

**Same object, different research failures.** On `macpherson_strut_housing`, Gemini fails at $L_0$ with a CadQuery/API error. Qwen reaches $L_0$, $L_1$, and $L_3$ with utilization ratio $0.0931$, but fails $L_2$ with score $0.25$, PIS $0.25$, ZSR $0.25$, and robustness $0.25$. The same overall failure therefore implies different fixes: CAD API grounding for Gemini, constraint-preserving generation for Qwen.

**Prompt specificity acts as an engineering specification channel.** For Qwen on `worm_gear_housing`, `level_1` executes but fails $L_1$ dimensionally, `level_2` fails $L_2$, and `level_3` becomes a co-author success with $L_2$ score $1.0$ and robustness $0.9167$. In engineering CAD, prompt detail is not cosmetic; it carries dimensions, relations, loads, and verification-relevant constraints.

## 7. Discussion and Limitations

**Benchmark interpretation.** CADEngBench is best read as a capability decomposition, not a monotone difficulty ladder. $L_0$ measures whether a model can construct a valid solid; $L_2$ measures whether the program remains editable under constraint-preserving perturbations; $L_3$ measures whether the nominal artifact satisfies an object-specific physics surrogate. Therefore $L_3$ can exceed $L_2$ without contradiction: a one-off geometry can be structurally acceptable while still failing as reusable parametric CAD. This distinction is the core benchmark claim.

**What the results imply.** The results argue against treating CAD generation as ordinary code generation plus rendering. CADCoder's $39/40$ $L_0$ score and $1/40$ overall score show that executable CadQuery is not enough. Conversely, the co-author successes show that frontier models can sometimes preserve dimensions, constraints, perturbation robustness, and physics validity simultaneously. Future systems should therefore optimize for constraint-aware generation, CAD API grounding, and verification feedback, not only syntactic validity or visual plausibility.

**Scope of evaluation.** CADEngBench deliberately uses programmatic CadQuery because scripts are re-executable, inspectable, and perturbable. This excludes sketch-native, assembly-native, and proprietary CAD workflows, but enables controlled scientific evaluation of generated design programs. The $L_3$ checks are closed-form engineering surrogates rather than FEA or CFD. They are intended as scalable, interpretable first-pass verification, not final industrial certification.

**Protocol limits.** The benchmark contains 40 objects by design: enough to expose unsolved tasks, unique solves, tier effects, and model-specific failures, but still small enough for audit and artifact inspection. Rankings should therefore be interpreted as evidence about this benchmark distribution, not universal model ordering. The main condition is single-sample, and the multigeneration ablation shows that this underestimates some frontier models' ceilings. Finally, the evaluated systems use heterogeneous deployment paths: four hosted APIs and one local CADCoder runtime. We disclose this because the paper evaluates end-to-end usable CAD generation, not provider-normalized serving conditions.

## 8. Conclusion

CADEngBench introduces a hierarchical test for whether AI-generated CAD programs can function as engineering design artifacts rather than one-shot geometry. The benchmark shows that executability is a weak endpoint: a specialized CAD model can nearly solve $L_0$ while failing the parametric and verification layers that determine reuse. The strongest systems are not those that only emit valid scripts, but those that preserve dimensions, constraints, perturbation behavior, and physics checks together. This makes $L_2$ parametric integrity the clearest current bottleneck and a concrete target for future CAD-generation research. More broadly, engineering design benchmarks should evaluate whether generated artifacts remain editable, auditable, and physically meaningful after generation; otherwise, they risk rewarding runnable shapes instead of engineering co-authorship.

## References

Chan, J. S., Chowdhury, N., Jaffe, O., Aung, J., Sherburn, D., Mays, E., Starace, G., Liu, K., Maksin, L., Patwardhan, T., Weng, L., and Madry, A. Mle-bench: Evaluating machine learning agents on machine learning engineering.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

de Moura, L. and Bjørner, N. *Z3: An Efficient SMT Solver*, pp. 337–340. Springer Berlin Heidelberg, 2008. doi:

10.1007/978-3-540-78800-3_24. URL https://doi.org/10.1007/978-3-540-78800-3_24.

Guan, Y., Wang, X., Xing, X., Zhang, J., Xu, D., and Yu, Q. Cad-coder: Text-to-cad generation with chain-of-thought and geometric reward, 2025. URL https://arxiv.org/abs/2505.19713.

Jain, N., Han, K., Gu, A., Li, W.-D., Yan, F., Zhang, T., Wang, S., Solar-Lezama, A., Sen, K., and Stoica, I. Livecodebench: Holistic and contamination free evaluation of large language models for code, 2024. URL https://arxiv.org/abs/2403.07974.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues?, 2024. URL https://arxiv.org/abs/2310.06770.

Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C., Manning, C. D., Ré, C., Acosta-Navas, D., Hudson, D. A., Zelikman, E., Durmus, E., Ladhak, F., Rong, F., Ren, H., Yao, H., Wang, J., Santhanam, K., Orr, L., Zheng, L., Yuksekgonul, M., Suzgun, M., Kim, N., Guha, N., Chatterji, N., Khattab, O., Henderson, P., Huang, Q., Chi, R., Xie, S. M., Santurkar, S., Ganguli, S., Hashimoto, T., Icard, T., Zhang, T., Chaudhary, V., Wang, W., Li, X., Mai, Y., Zhang, Y., and Koreeda, Y. Holistic evaluation of language models, 2023. URL https://arxiv.org/abs/2211.09110.

Zhang, L., Le, B., Akhtar, N., Lam, S.-K., and Ngo, T. Large language models for computer-aided design: A survey, 2026. URL https://arxiv.org/abs/2505.08137.

## A. Dataset and Benchmark Suite

Each CADEngBench task is an object contract with four synchronized files: `prompts.json` defines the model-facing prompt ladder, `spec.json` defines semantic metadata and physics parameters, `constraints.json` defines machine-checkable dimensions and relations, and `ground_truth.py` defines executable reference CadQuery. The benchmark contains 40 objects across Tier 1/2/3 counts of $11/22/7$, 12 engineering domains, and 17 physics tags.

## B. Protocol, Evaluator, and Validity

The main condition evaluates five models on all 40 objects with `level_3` prompts, one generation per object, and all four layers. The prompt ablation uses a 12-object slice with `level_1`, `level_2`, and `level_3` for all five models. The multigeneration ablation uses three samples per object for Claude Sonnet 4.6, GPT-5.1, and CADCoder on the same slice.

**$L_0$ runtime.** The compiler stage executes the extracted Python script in a controlled CadQuery/OCC subprocess, requires a top-level `result`, accepts CadQuery `Workplane` or shape-like objects, and records bbox, volume, face/edge/vertex counts, Euler diagnostics, open-edge count, non-manifold diagnostics, scalar globals, parameter dictionaries, timeout failures, kernel failures, CAD API failures, and empty-geometry failures.

**$L_1$ measurement.** The geometry stage applies object-aware bbox policies, tolerance diagnostics, axis permutation handling, and feature extractors. It evaluates relational predicates from `constraints.json`, while script-intelligence diagnostics such as parameter-dictionary presence, named-variable coverage, formula count, and compression distance are logged for analysis rather than replacing geometric checks.

**$L_2$ editability.** The parametric stage detects whether a script uses dictionaries, named variables, or value-matched literals; samples perturbations over $\theta'_j \in [0.75\theta_j, 1.25\theta_j]$ using a deterministic Morris design; substitutes values; reruns $L_0/L_1$; and computes sample robustness, behavioral propagation, and solver-backed preservation.

**$L_3$ physics.** The physics stage dispatches by `physics_tag`, extracts common geometry, runs the tag-specific closed-form verifier, and reports a utilization ratio. Co-author status requires $L_3$ pass plus PIS and ZSR thresholds; a nominal physics pass without strong parametric integrity is labeled *tool*.

All ground-truth scripts execute through the same evaluator. The only ground-truth $L_2$ exception is `disc_brake_rotor`, with $S_{L_2} = 0.25$, PIS $= 0.0$, and ZSR $= 0.5$; it is retained as an audit item rather than hidden. The $L_3$ registry covers all 17 dataset physics tags, with no missing dataset tags.

## C. Supplementary Diagnostics and Ablations

The main paper reports the core ranking, $L_2$ decomposition, first-failure groups, and ablations. The supplement avoids repeating object-level pass matrices visually; those machine-readable matrices are preserved in CSV/JSON artifacts. Figures 7–9 provide one-column diagnostic views, while Algorithms 2–3 summarize the two engineering-specific evaluator stages.

*Table 6.* Object-contract schema. The model sees only a prompt; the remaining files define hidden measurement, perturbation, and verification contracts.

| File | Main contents | Evaluator use | Why it matters |
|---|---|---|---|
| `prompts.json` | `level_1`, `level_2`, `level_3` natural-language specifications | generation input only; prompt ablations swap this file view | separates task information from hidden scoring rules |
| `spec.json` | object id, tier, domain, physics tag, material/load parameters, nominal values | $L_3$ dispatch, physics constants, load case metadata | makes physics verification object-aware rather than generic mesh scoring |
| `constraints.json` | dimensional targets, relational predicates, perturbable variables, dependent formulas | $L_1$ measurements, $L_2$ perturbation expectations, Z3 checks | encodes design intent and constraint propagation under edits |
| `ground_truth.py` | executable CadQuery reference using the same object contract | ground-truth validation and qualitative STL exports | audits that the benchmark object is executable, measurable, and physically valid |

*Table 7.* Compact CADEngBench object catalog. Each half-row lists object, tier, domain, and physics tag.

| Object | Tier | Domain | Tag | Object | Tier | Domain | Tag |
|---|---|---|---|---|---|---|---|
| `arch_bridge_segment` | T3 | civil | arch | `pressure_relief_valve_body` | T2 | mechanical | pressure_vessel |
| `cantilever_beam` | T1 | mechanical | beam | `rectangular_fin_array` | T1 | thermal | heat_transfer |
| `cantilever_bracket` | T1 | mechanical | bracket | `rectangular_plate_hole` | T1 | mechanical | plate |
| `centrifugal_pump_impeller` | T2 | fluid | fluid | `rectangular_weir` | T2 | fluid | fluid |
| `circular_column` | T1 | mechanical | column | `shell_and_tube_heat_exchanger` | T3 | thermal | heat_exchanger |
| `circular_footing` | T2 | structural_foundation | foundation | `simply_supported_beam` | T1 | mechanical | beam |
| `connecting_rod` | T3 | mechanical | column | `spur_gear_tooth` | T2 | mechanical_gear | gear |
| `crankshaft_journal` | T2 | automotive | shaft | `stepped_shaft` | T2 | mechanical | shaft |
| `disc_brake_rotor` | T2 | mechanical | thermal | `t_section_beam` | T2 | mechanical | beam |
| `finned_heat_sink` | T1 | thermal | heat_transfer | `t_section_bridge_girder` | T2 | civil | beam |
| `flanged_pipe_joint` | T2 | mechanical | pressure_vessel | `t_slot_nut` | T2 | manufacturing | thread |
| `heat_exchanger_tube` | T2 | thermal_heat_exchange | pressure_vessel | `thick_walled_cylinder` | T2 | mechanical_pressure_vessel | pressure_vessel |
| `hollow_cylinder` | T1 | mechanical_pressure_vessel | pressure_vessel | `thin_walled_tube` | T2 | mechanical | pressure_vessel |
| `hollow_rectangular_beam` | T2 | mechanical | beam | `trapezoidal_cross_section_beam` | T2 | mechanical | beam |
| `hollow_rectangular_pier` | T3 | civil | column | `trapezoidal_retaining_wall` | T2 | civil | retaining |
| `i_beam_section` | T2 | mechanical | beam | `venturi_flow_meter` | T2 | fluid | fluid |
| `injection_mould_core_pin` | T2 | manufacturing | column | `venturi_meter` | T1 | fluid | fluid |
| `l_bracket` | T1 | structural | bracket | `wheel_rim_cross_section` | T2 | automotive | pressure_vessel |
| `leaf_spring` | T1 | mechanical | spring | `wing_spar_box_section` | T3 | aerospace | torsion |
| `macpherson_strut_housing` | T3 | automotive | bracket | `worm_gear_housing` | T3 | mechanical | gear |

*Table 8.* Benchmark-suite distribution and diverse STL export metadata. Domain/tag counts describe the full 40-object suite; STL rows describe the 10 rendered references in Figure 6.

| Domain | Count | Physics tag | Count | STL object | Domain/tag | Bbox (mm) | Volume |
|---|---|---|---|---|---|---|---|
| mechanical | 17 | beam | 7 | `arch_bridge_segment` | civil/arch | $6000 \times 300 \times 1640$ | 288.28M |
| civil | 4 | pressure_vessel | 7 | `circular_footing` | structural_foundation/foundation | $2201 \times 2200 \times 850$ | 2100.15M |
| fluid | 4 | column | 4 | `crankshaft_journal` | automotive/shaft | $49 \times 49 \times 34$ | 44.3k |
| automotive | 3 | fluid | 4 | `finned_heat_sink` | thermal/heat_transfer | $60 \times 60 \times 20$ | 34.2k |
| thermal | 3 | bracket | 3 | `l_bracket` | structural/bracket | $40 \times 80 \times 60$ | 66.3k |
| manufacturing | 2 | gear | 2 | `shell_and_tube_heat_exchanger` | thermal/heat_exchanger | $154 \times 154 \times 1548$ | 6.77M |
| mechanical_pressure_vessel | 2 | heat_transfer | 2 | `spur_gear_tooth` | mech._gear/gear | $66 \times 66 \times 30$ | 37.7k |
| aerospace | 1 | shaft | 2 | `t_slot_nut` | manuf./thread | $30 \times 27 \times 12$ | 7.6k |
| mechanical_gear | 1 | arch | 1 | `venturi_flow_meter` | fluid/fluid | $90 \times 90 \times 176$ | 638.2k |
| structural | 1 | foundation | 1 | `wing_spar_box_section` | aerospace/torsion | $2000 \times 200 \times 80$ | 4.06M |
| structural_foundation | 1 | heat_exchanger | 1 | – | – | – | – |
| thermal_heat_exchange | 1 | plate, retaining, spring, thermal, thread, torsion | 1 each | – | – | – | – |

*Table 9.* Run protocol, preserved artifacts, evaluator dependencies, and ground-truth validity in one place.

| Block | Scope / dependency | Preserved output | Validation note |
|---|---|---|---|
| Main run | 5 models $\times$ 40 objects $\times$ 1 generation | raw response, script, CAD, JSON reports | `level_3` prompt condition |
| Prompt ablation | 5 models $\times$ 12 objects $\times$ 3 prompt levels | same artifact trail | tests specification dependence |
| Multigeneration | 3 models $\times$ 12 objects $\times$ 3 samples | same artifact trail | tests sampling ceiling |
| $L_0$ | no dependency | valid solid $G(x)$ and topology diagnostics | ground truth $40/40$ |
| $L_1$ | depends on $L_0$ | measured namespace $\hat{\theta}$ | ground truth $40/40$ |
| $L_2$ | depends on $L_0$ | $S_{L_2}$, PIS, ZSR, robustness | ground truth $39/40$ |
| $L_3$ | depends on $L_0$ | utilization ratio and autonomy label | ground truth $40/40$ |
| Validity audit | design docs, static checks, case studies | audit notes and summaries | 5 documents, 6 findings, 4 case studies |

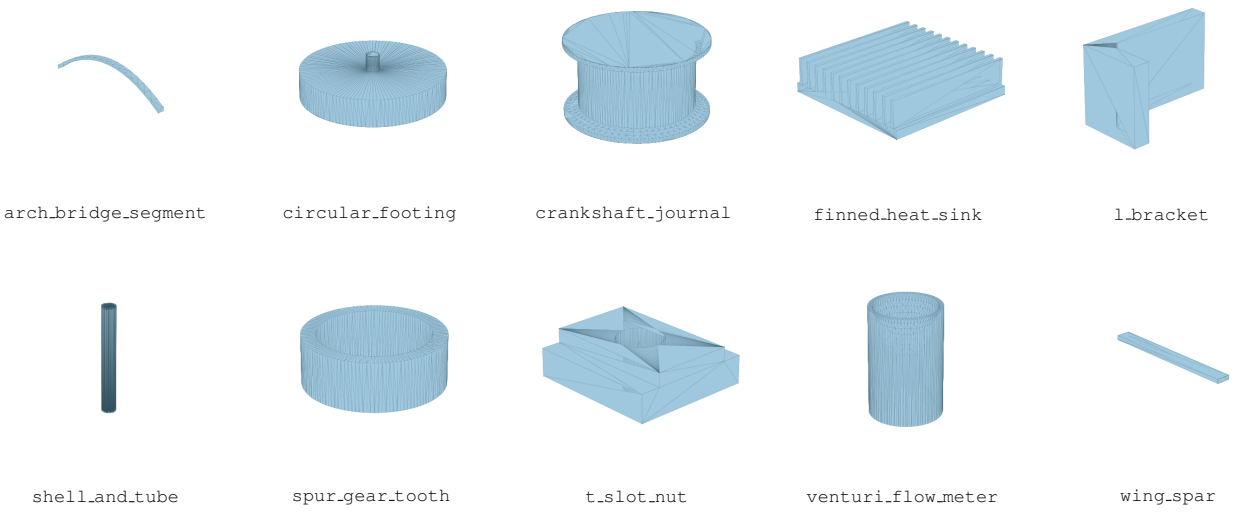



*Figure 6.* Ground-truth STL render gallery for a diverse 10-object subset. These renders demonstrate benchmark breadth across scale, geometry, domain, and physics tag; they are not used for scoring.

*Table 10.* Implemented $L_3$ physics tags, dataset coverage, and governing-equation families. Each verifier returns a nominal utilization ratio or analogous pass/fail scalar used by the $L_3$ report.

| Physics tag | $n$ | Representative objects | Dataset scope | Governing-equation family |
|---|---|---|---|---|
| `arch` | 1 | `arch_bridge_segment` | Parabolic arch bridge segment under symmetric distributed loading. | $H = wL^2/(8f)$; crown compression $\sigma = H/A$. |
| `beam` | 7 | cantilever; I/T-section; trapezoidal | Cantilever, simply supported, hollow box, T-section, I-section, and trapezoidal beam sections. | Euler–Bernoulli bending/deflection; $\sigma = Mc/I$, $\delta \propto FL^3/EI$. |
| `bracket` | 3 | cantilever bracket; strut housing | Plate/web-root brackets and tubular strut-housing cantilever sections. | Combined bending/shear von Mises stress, $\sigma_{\mathrm{vm}} = \sqrt{\sigma_b^2 + 3\tau^2}$. |
| `column` | 4 | circular column; hollow pier | Solid circular columns, core pins, connecting rods, and hollow piers. | Euler buckling or axial-flexural compression, $\sigma = P/A + Mc/I$. |
| `fluid` | 4 | venturi; weir; pump impeller | Venturi meters, pump impellers, and sharp-crested rectangular weirs. | Incompressible discharge, Euler pump head, or open-channel weir flow. |
| `foundation` | 1 | `circular_footing` | Concentric circular spread footing under service load. | Shallow bearing pressure, $q = P/A$. |
| `gear` | 2 | spur gear; worm housing | Tooth-bending and housing-wall checks. | Lewis tooth bending or housing wall bending, $\sigma = Mc/I$. |
| `heat_exchanger` | 1 | shell-and-tube HX | Shell-and-tube heat-exchanger performance surrogate. | LMTD or NTU/effectiveness relation. |
| `heat_transfer` | 2 | `finned_heat_sink`; `rectangular_fin_array` | Rectangular fin arrays under natural convection. | Straight-fin efficiency, $\eta = \tanh(mL_c)/(mL_c)$. |
| `plate` | 1 | `rectangular_plate_hole` | Rectangular plate with through-hole and finite-width correction. | Net-section or hole-corrected plate stress. |
| `pressure_vessel` | 7 | thick cylinder; flange; wheel rim | Thin-wall and thick-wall cylindrical shells under internal pressure. | Membrane stress or Lamé thick-wall hoop stress. |
| `retaining` | 1 | `trapezoidal_retaining_wa` | Gravity retaining wall under earth pressure. | Overturning stability and base-pressure utilization. |
| `shaft` | 2 | `crankshaft_journal`; `stepped_shaft` | Stepped and journal shafts under bending/torsion. | Von Mises and fatigue-style bending–torsion combination. |
| `spring` | 1 | `leaf_spring` | Laminated semi-elliptic leaf spring. | Closed-form spring stress, stiffness, and deflection. |
| `thermal` | 1 | `disc_brake_rotor` | Disc-brake rotor thermal-stress surrogate. | Application-specific thermal energy or thermoelastic relation. |
| `thread` | 1 | `t_slot_nut` | Thread engagement and root checks for a T-slot nut. | Engagement shear area and pull-out/thread-root stress. |
| `torsion` | 1 | `wing_spar_box_section` | Closed thin-walled wing-spar box section. | Bredt–Batho shear-flow torsion. |

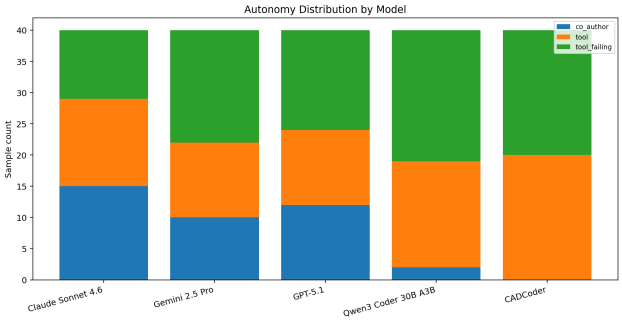



*Figure 7.* Autonomy label distribution by model. Co-author outputs require more than executable geometry; they must also satisfy the benchmark's editability and physics criteria.

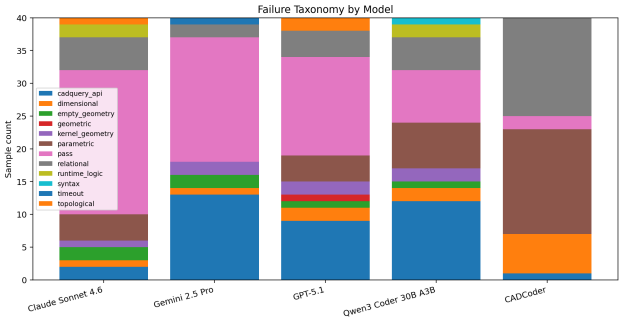



*Figure 8.* Grouped first-failure taxonomy by model. The distribution separates early construction failures from specification, parametric, and physics failures.

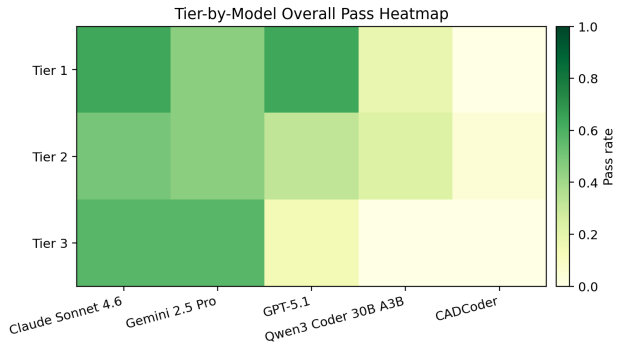



*Figure 9.* Tier-level pass-rate summary. The tier view complements the paired bootstrap analysis by showing where difficulty sensitivity enters.

**Algorithm 2** $L_2$ parametric-integrity evaluation.

**Require:** generated script $x$, object contract $q_o$
1: detect substitution mode: dictionary, variable, or literal fallback
2: sample $\theta'_j \in [0.75\theta_j, 1.25\theta_j]$
3: **for** each perturbation sample **do**
4: derive expected dependent namespace
5: skip infeasible samples under $q_o$
6: substitute sampled values into $x$
7: rerun $L_0$/$L_1$ and record failures
8: **end for**
9: compute PIS and ZSR from propagation checks $S_{L_2}$, robustness, counterexamples

**Algorithm 3** $L_3$ physics dispatch.

**Require:** generated solid $G(x)$, `spec.json`, `constraints.json`
1: **if** $L_0$ did not produce a valid solid **then**
skipped physics report
2: **end if**
3: normalize CadQuery/OCC shape
4: extract common geometry and load parameters
5: dispatch evaluator by `physics_tag`
6: compute utilization ratio and pass/fail verdict
7: combine verdict with $L_2$ metrics for autonomy label

**Reproducibility and artifact release.** All paper runs are organized under `results/experiments/` with the `paper_` prefix. Each run preserves prompts, raw model responses, extracted scripts, generated CAD artifacts, evaluator JSON reports, and aggregate CSV/JSON summaries. The appendix is supplementary: every core claim used for evaluation, ranking, and interpretation appears in the main paper, while this section exposes the implementation and artifact trail needed to audit the claims.

Prompt specificity tests whether failures are due to missing task information; repeated sampling tests whether a model has latent capability hidden by a single sample. The two ablations separate these effects: frontier hosted models recover with better prompts and sometimes with more samples, while CADCoder remains blocked by parametric integrity.

*Table 11.* Object-level paired bootstrap intervals. Means are object-level pass rates.

| Comparison | Overall pass A | B | Diff. | 95% CI | $L_2$ pass A | B | Diff. | 95% CI |
|---|---|---|---|---|---|---|---|---|
| Claude vs. Gemini | 0.550 | 0.475 | 0.075 | $[-0.125, 0.275]$ | 0.575 | 0.500 | 0.075 | $[-0.100, 0.275]$ |
| Claude vs. GPT-5.1 | 0.550 | 0.375 | 0.175 | $[0.025, 0.325]$ | 0.575 | 0.450 | 0.125 | $[-0.050, 0.300]$ |
| Claude vs. Qwen3 | 0.550 | 0.175 | 0.375 | $[0.200, 0.550]$ | 0.575 | 0.200 | 0.375 | $[0.175, 0.575]$ |
| Claude vs. CADCoder | 0.550 | 0.025 | 0.525 | $[0.375, 0.700]$ | 0.575 | 0.050 | 0.525 | $[0.350, 0.700]$ |
| Gemini vs. GPT-5.1 | 0.475 | 0.375 | 0.100 | $[-0.050, 0.250]$ | 0.500 | 0.450 | 0.050 | $[-0.125, 0.200]$ |
| Gemini vs. Qwen3 | 0.475 | 0.175 | 0.300 | $[0.175, 0.425]$ | 0.500 | 0.200 | 0.300 | $[0.125, 0.451]$ |
| Gemini vs. CADCoder | 0.475 | 0.025 | 0.450 | $[0.300, 0.600]$ | 0.500 | 0.050 | 0.450 | $[0.300, 0.600]$ |
| GPT-5.1 vs. Qwen3 | 0.375 | 0.175 | 0.200 | $[0.050, 0.350]$ | 0.450 | 0.200 | 0.250 | $[0.075, 0.425]$ |
| GPT-5.1 vs. CADCoder | 0.375 | 0.025 | 0.350 | $[0.200, 0.500]$ | 0.450 | 0.050 | 0.400 | $[0.250, 0.550]$ |
| Qwen3 vs. CADCoder | 0.175 | 0.025 | 0.150 | $[0.025, 0.275]$ | 0.200 | 0.050 | 0.150 | $[0.025, 0.275]$ |

*Table 12.* Supplementary ablation tables grouped into one float. Prompt rows report pass counts out of 12; multigeneration rows report object-level pass@1/pass@3 on the same 12-object slice.

| Model | Prompt | $L_0$ | $L_1$ | $L_2$ | $L_3$ | Overall |
|---|---|---|---|---|---|---|
| Claude Sonnet 4.6 | `level_1` | 11/12 | 2/12 | 0/12 | 8/12 | 0/12 |
| Claude Sonnet 4.6 | `level_2` | 10/12 | 8/12 | 6/12 | 9/12 | 6/12 |
| Claude Sonnet 4.6 | `level_3` | 12/12 | 9/12 | 8/12 | 9/12 | 5/12 |
| Gemini 2.5 Pro | `level_1` | 5/12 | 3/12 | 1/12 | 2/12 | 0/12 |
| Gemini 2.5 Pro | `level_2` | 5/12 | 4/12 | 3/12 | 4/12 | 3/12 |
| Gemini 2.5 Pro | `level_3` | 7/12 | 5/12 | 6/12 | 6/12 | 5/12 |
| GPT-5.1 | `level_1` | 8/12 | 1/12 | 0/12 | 6/12 | 0/12 |
| GPT-5.1 | `level_2` | 10/12 | 7/12 | 6/12 | 9/12 | 5/12 |
| GPT-5.1 | `level_3` | 9/12 | 7/12 | 6/12 | 8/12 | 5/12 |
| Qwen3 Coder 30B A3B | `level_1` | 5/12 | 2/12 | 1/12 | 5/12 | 1/12 |
| Qwen3 Coder 30B A3B | `level_2` | 10/12 | 7/12 | 4/12 | 7/12 | 3/12 |
| Qwen3 Coder 30B A3B | `level_3` | 8/12 | 6/12 | 6/12 | 8/12 | 5/12 |
| CADCoder | `level_1` | 11/12 | 4/12 | 0/12 | 5/12 | 0/12 |
| CADCoder | `level_2` | 10/12 | 7/12 | 0/12 | 6/12 | 0/12 |
| CADCoder | `level_3` | 12/12 | 8/12 | 0/12 | 8/12 | 0/12 |

| Model | p@1 | p@3 | $L_2$@1 | $L_2$@3 |
|---|---|---|---|---|
| Claude | 5/12 | 9/12 | 8/12 | 11/12 |
| GPT-5.1 | 5/12 | 6/12 | 5/12 | 9/12 |
| CADCoder | 0/12 | 0/12 | 0/12 | 0/12 |

*Table 13.* Artifact-backed case-study manifest. UR denotes utilization ratio.

| Case | Model / object | Stage | $S_{L_2}$ | PIS | ZSR | UR | Diagnostic point |
|---|---|---|---|---|---|---|---|
| Hard success | Claude / `worm_gear_housing` | pass | 1.00 | 1.00 | 1.00 | 0.384 | Tier 3 co-author success with high perturbation robustness. |
| Hard success | GPT-5.1 / `hollow_rectangular_pier` | pass | 1.00 | 1.00 | 1.00 | 0.557 | Hard civil object solved despite weaker aggregate Tier 3 performance. |
| Specialist collapse | CADCoder / `thick_walled_cylinder` | $L_2$ | 0.25 | 0.00 | 0.50 | 0.117 | Executable and nominally useful geometry without reusable parametric structure. |
| Construction failure | Gemini / `macpherson_strut_housing` | $L_0$ | 0.00 | 0.00 | 0.00 | – | Early CadQuery/API failure before downstream engineering checks. |
| Parametric failure | Qwen3 / `macpherson_strut_housing` | $L_2$ | 0.25 | 0.25 | 0.25 | 0.093 | Same object reaches execution but fails constraint-preserving editability. |
| Prompt recovery | Qwen3 / `worm_gear_housing` | pass | 1.00 | 1.00 | 1.00 | 0.384 | `level_1` fails dimensionally; `level_3` recovers to co-author status. |