Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Special Track on AI4Tech: AI Enabling Critical Technologies

Physics-in-the-Loop: A Hybrid Agentic Architecture for Validated CAD Engineering Design

Elias Berger[1,2] and **Muhammad Usama**[3,4] and **Jan Mehlstäubl**[2] and **Bernhard Saske**[1] and **Kristin Paetzold-Byhain**[1]
[1]Dresden University of Technology, Dresden, Germany
[2]MAN Truck & Bus SE, Munich, Germany
[3]German Research Center for Artificial Intelligence, Kaiserslautern, Germany
[4]RPTU Kaiserslautern-Landau, Germany
{elias.berger, bernhard.saske, kristin.paetzold-byhain}@tu-dresden.com,
jan.mehlstaeubl@digitalhub.man, muhammad.usama@dfki.de

Abstract

Large Language Models (LLMs) can generate Computer-Aided Design (CAD), yet lack physical comprehension required for reliable engineering design. Instead of attempting to implicitly learn physical laws from data, we propose a Hybrid Agentic-Physical Architecture that embeds validated knowledge-based engineering tools directly into the decision-making loop of autonomous AI agents. In this framework, engineering design is formulated as a closed-loop, sequential decision-making process guided by explicit physical verification. Based on a load case, dedicated agents iteratively plan, generate, evaluate, and revise engineering designs using knowledge-based tools as a feedback signal. We introduce a benchmark dataset and metrics for assessing functional validity in generative CAD. Our system generates more complex and physically verified designs, with a 4.2× increase in structural complexity and improving compile rate by 3.5% compared to similar agentic methods. The benchmark data is available on the project page[1].

1 Introduction

Generative artificial intelligence (AI) has recently gained momentum in engineering design [Zhang *et al.*, 2025; Steininger *et al.*, 2025; Regassa Hunde and Debebe Woldeyohannes, 2022]. Notwithstanding this progress, transferring these methods into real-world engineering practice, where reliable and validated structural integrity is essential, remains largely an open challenge. [Berger *et al.*, 2025; Preintner *et al.*, 2025].

LLM-based generative methods enable the synthesis of parametric CAD models from textual and multimodal inputs [Xu *et al.*, 2025]. However, existing generative CAD systems are predominantly assessed using geometric similarity metrics, which fail to capture functional validity, load-bearing capacity, or other engineering objectives [Berger *et al.*, 2026a]. Generated designs appear visually plausible but lack guarantees of functional correctness or technical feasibility. Recent efforts have attempted to improve reliability and complexity through iterative feedback using Vision Language Models (VLM) to visually verify 2D renders of CAD outputs [Alrashedy *et al.*, 2025; Preintner *et al.*, 2025]. While effective for shape fidelity, visually-evaluated systems do not verify the structural integrity of the generated parts and produce very simple geometries. These approaches use the DeepCAD dataset [Wu *et al.*, 2021], which contains mostly very simple geometries with no labels relevant for engineering design.

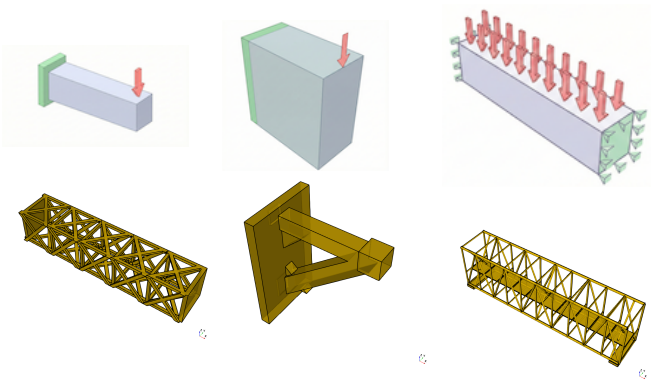

Figure 1: Generated CAD design examples. Top row: Input load cases defining design space, boundary conditions and forces. Bottom row: Resulting geometries satisfying the physical constraints.

In this paper, we formulate generative CAD as a physics-constrained engineering problem. We propose an Agentic CAD System that uses VLMs to automate mechanical design through an iterative "Generate-Simulate-Refine" loop. Unlike previous works that rely on visual feedback, our method integrates a validated knowledge-based tool into the feedback loop. Our agents receive feedback from physics-based tools to iteratively enhance the output to be not only geometrically valid but structurally sound. We do not train VLMs, but make use of the multi-task in-context capabilities of VLMs.

[1]https://github.com/rapidcad/structbench

Thus our approach does not rely on large annotated datasets. Further, we define new metrics and publicize a benchmark dataset for evaluating functional validity in generative CAD. Our method does not depend on existing training datasets, as we leverage the multi-task capabilities of LLMs and VLMs. This allows us to generate designs of high complexity, exceeding prior works by a factor of three in geometric complexity (examples see Figure 1).

Our contributions are as follows:

1. A hybrid multi-agent architecture integrating physics-in-the-loop active decision signal that generates CAD design with exceeding complexity.
2. Systematic comparison of generative, agentic, and hybrid AI paradigms and empirical evidence that physics-guided agents improve reliability and physical validity.
3. A novel benchmark for functional CAD generation under load bearing requirements, made publicly available.

2 Related Work

2.1 LLMs for CAD and Engineering Design

DeepCAD [Wu *et al.*, 2021] is the initial paper that pioneered deep learning methods for CAD generation using a design history. Since then several studies demonstrate the ability of LLMs to generate parametric CAD models from textual [Khan *et al.*, 2024b; Wang *et al.*, 2025b; Lv and Bao, 2025; Wang *et al.*, 2025a; Govindarajan *et al.*, 2026; Usama *et al.*, 2025; Berger *et al.*, 2026b], image [T. Chen *et al.*, 2025; Alam and Ahmed, 2025], point cloud [Dupont *et al.*, 2024; Khan *et al.*, 2024a], or multi-modal [Doris *et al.*, 2025; Xu *et al.*, 2025; Kolodiazhnyi *et al.*, 2025] inputs.

Agentic and iterative generation processes have been recently proposed. Preintner *et al.* integrates an evolutionary optimization loop to iteratively improve generated designs based on visual feedback. Ocker *et al.* propose a Multi-Agent System driven by a VLM to automate CAD model generation by mirroring the structure of human engineering teams with specialized agents. Alrashedy *et al.* introduces CADCodeVerify, a method that employs commercial VLMs and prompting to generate CAD code and uses visual tests to verify if the CAD object matches the intended shape.

A shared challenge of these methods is the reliance on a limited training dataset. Public datasets such as DeepCAD [Wu *et al.*, 2021], Fusion360 [Willis *et al.*, 2021], and CAD-Parser [Zhou *et al.*, 2023] contain only a few ten thousand CAD models with little complexity, which is small compared to datasets in other domains.

Other approaches operate on different representations such as Boundary Representation (B-Rep) [Jayaraman *et al.*, 2023; Lambourne *et al.*, 2021] or meshes [Nash *et al.*, 2020] but do not produce CAD models with a design history.

Further, the evaluation of these generative CAD methods has so far been primarily based on geometric similarity metrics such as Chamfer Distance, Intersection-over-Union (IoU), or Normal Consistency. While optimizing replication capabilities, the metrics lack consideration of real-world functional requirements like load-bearing capacity [Preintner *et al.*, 2025; He *et al.*, 2025]. Consequently, this research area would benefit from new data and metrics.

2.2 Agentic AI and Multi-Agent Systems

Agentic AI commonly describes systems in which one or more LLMs are embedded within an execution loop. Agentic systems allow for task distribution, observation of intermediate outcomes, and iterative refinement towards a specified objective. [Plaat *et al.*, 2025; Huang, 2024]. In contrast to single-turn text generation, agentic architectures tightly integrate reasoning with interaction in an external environment. This enables using feedback to improve task completion and reduce hallucinations without requiring updates to model parameters. [Yao *et al.*, 2023; Shen, 2024].

2.3 Knowledge-based Engineering and Hybrid AI

Knowledge-based engineering tools are explicit physical models and formalized design knowledge that, compared to LLMs, have already reached a very mature state [Herrmann *et al.*, 2021]. Established tools such as topology optimization and FEA use physics-based objectives and constraints to reliably produce functionally optimized designs, albeit within restricted domains and no learning on past data. The fundamental gap between generative creativity and validated engineering tools motivates hybrid approaches that combine data-driven AI with knowledge-based tools.

3 Problem Formulation

With this work we aim to contribute towards AI for engineering design – a discipline that focuses on the creation of technical parts that fulfill specific functional requirements [Pahl *et al.*, 2007]. Engineering design commonly follows an iterative process, where software tools such as CAD and FEA are used to design and validate artifacts digitally [Hirz *et al.*, 2011]. Besides the geometric appearance of an artifact in engineering design its structural and functional performance under physical loads is of key importance [Schulte *et al.*, 1993]. The input for the problem is a structured load case defining how a part is constrained at fixed supports, the forces acting upon it, and the design space within which it is defined. The task is to generate a CAD model that satisfies the load case.

We use an established PyTorch-based FEA solver[2] to evaluate the structural performance and the objective of the AI system is to achieve a safety factor between 2.0 and 5.0, consistent with engineering practice [Budynas and Nisbett, 2020] while minimizing the volume of the part. We use gmsh[3] for meshing with optimization enabled for robustness. We set the material properties to match Aluminum 8081.

4 Load Case Data

We formulate the engineering design problem in terms of load bearing requirements that we formalize in load cases. Existing CAD datasets are not sufficient as they are not annotated with load case information necessary for physics-based validation. To this end, we construct a novel benchmark dataset

[2]https://github.com/meyer-nils/torch-fem

[3]https://gmsh.info

```
{
  "meta": { "problem_id": "ARCH_BRIDGE", ...
      },
  "design_domain": {
    "units": "mm",
    "bounds": { "x_max": 1000.0, ... }
  },
  "spatial_selectors": [
    { "id": "support_left", "query": {
       "x_min": 0.0, "x_max": 50.0, ...
    } },
    ...
  ],
  "boundary_conditions": [
    { "spatial_selector_id": "support_left",
      "type": "fixed_displacement",
      "dof_lock": { ... } }
  ],
  "loads": [
    { "spatial_selector_id": "deck_surface",
      "type": "distributed_force",
      "magnitude_newtons": ...,
      "direction": ... },
  ]
}
```

Figure 2: Condensed excerpt of a load case JSON definition. `design_domain` specifies the design space. `spatial_selectors` define regions of interest for applying `boundary_conditions` and `loads` to the selected regions. `dof_lock` refers to degrees of freedom locked at fixed supports.

comprising 20 representative load cases derived from standard mechanical design tasks [Mahamid *et al.*, 2020]. Together with mechanical engineers we crafted the load cases to cover common challenges such as non-concave design spaces and internal holes. Each load case specifies fixed supports, applied forces, and explicit design space constraints. To evaluate generalization under varying constraints, we modify each load case using five distinct geometric scales and five force magnitude scales. As structural stiffness scales nonlinearly with geometric size, variations in scale and loading induce distinct mechanical problems. In total, the benchmark comprises 500 unique load case configurations. Representative samples are illustrated in the left column of Figure 4. An example of the JSON schema is shown in Figure 2. We make the load case data available as JSON files and the CAD models as Python code.

5 System Architecture

We propose a multi-agent framework built on LangGraph that coordinates specialized reasoning agents to enact an iterative "Generate-Simulate-Refine" engineering loop. This architecture formalizes the human design process as a graph of stateful nodes (current plan, CAD code, validator outputs) with directed cyclic execution. Feedback from downstream agents and physics-based tools directly informs upstream decision-making (see Figure 3).

5.1 Core Components

The system is composed of four agents, orchestrated by a central control graph:

Planner Agent. This agent acts as the semantic bridge between user intent and geometric modeling. It decomposes load bearing requirements into a specific, step-by-step modeling plan. As input it receives the load case JSON and a visualization of the design space with boundary conditions and forces.

CAD Engineer Agent. The CAD Engineer consumes the structured plan and translates it into executable Python code using the CadQuery library. It ensures that the generated script is syntactically correct and topologically consistent. The CAD engineer agents receives feedback from the geometry reviewer to fix code and modelling errors.

Geometry Reviewer Agent. Receives as input the plan and renders of the CAD object from different angles. Acting as the critic, it checks if the geometry matches expectations, if the proposed part connects forces and constraints, is meshable, and enforces design space restrictions. It has physics-based tools for verifying connectivity and design space compliance. Its feedback is used to fix modelling errors.

Structural Reviewer Agent. Executes physics-based FEA to assess structural performance, evaluates whether the safety factor is within a specified target range, and identifies stress hotspots or over-engineering. Its feedback is used to refine the plan.

If any physics-based evaluation fails, the Orchestrator routes feedback to the Planner and CAD Engineer for refinement; agents may inspect but cannot modify deterministic evaluation results. A solution is considered valid if and only if all deterministic geometric and physics-based checks pass. The prompt templates are available on the project page.

6 Experimental Setup

We evaluate all models in an inference-only setting without any parameter training. A single disjoint load-case example is provided for in-context learning. The in-context example is excluded from the evaluation set and the model context is cleared after each run to prevent cross-instance information leakage. To assess the capabilities of the proposed agentic framework, we designed a benchmarking protocol that evaluates the system's ability to autonomously generate functional mechanical components from high-level specifications. We benchmark the performance of the architecture using four state-of-the-art VLMs as agents: Anthropic Claude 4.5 Sonnet, Anthropic Claude 4.5 Opus, Google Gemini 3 Pro, and Google Gemini 3 Flash. The task is design a component given the load case and achieve a safety factor within the target range while respecting the given design space. The agents are granted a maximum of 10 iterations per problem to converge on a valid solution. We perform three independent runs per load case and model to account for stochasticity in the generation process.

6.1 Evaluation Metrics

In contrast to previous publications, we do not make use of geometry-based metrics, because in engineering design the

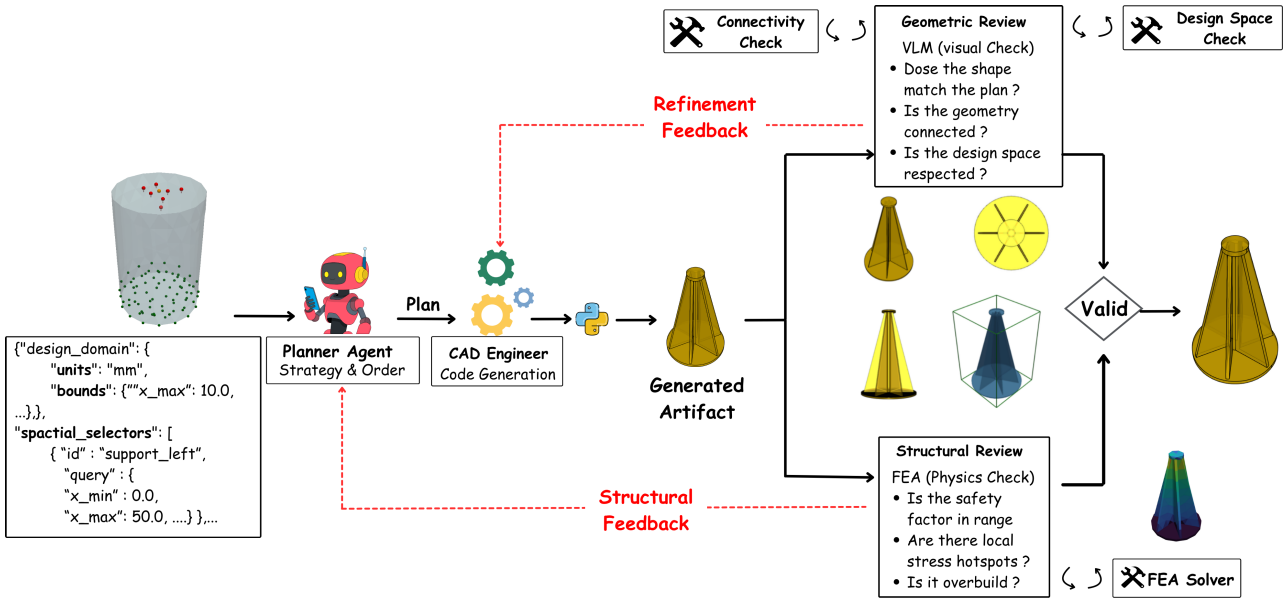

Figure 3: Hybrid Agentic-Physical Architecture. The system processes structured load cases (left) through a multi-agent generation loop. The generated geometry (center) is subjected to parallel validation: a Vision-Language Model evaluates shape fidelity, while Finite Element Analysis rigorously verifies structural performance. Feedback from both streams guides the CAD Engineer in iterative refinement.

functionality is of higher importance than geometric similarity to a reference design [Schulte *et al.*, 1993]. Therefore, we introduce a new set of quantitative metrics to evaluate performance in reliability and design quality. These metrics are closely aligned with real-world engineering utility and are computed deterministically outside of VLMs.

Reliability. Geometry Generation Success Rate ($\mathbf{R}_1$): The percentage of generated CAD scripts that execute without geometric errors (e.g., extruding open sketches). **Meshing Success Rate ($\mathbf{R}_2$):** Fraction of geometries that can be successfully meshed. Modeling errors like non-manifold edges prevent meshing. **FEA Success Rate ($\mathbf{R}_3$):** The percentage of generated models that successfully pass the finite element solving stages. Errors like disconnected geometry cause failure.

Design quality. Safety Factor ($\mathbf{DQ}_1$): The minimum structural safety factor (yield strength / max stress). Higher values indicate robustness; excessive values indicate over-engineering. **Structural Efficiency Ratio ($\mathbf{DQ}_2$):** $SFR = \text{Safety Factor}/\text{Volume}$. Identifies designs optimizing the strength-weight trade-off. **Number of Faces ($\mathbf{DQ}_3$):** Number of unique faces in the B-Rep geometry, proxying geometric complexity. This metric captures the trade-off between expressiveness and complexity. **Design Space Violation Rate ($\mathbf{DQ}_4$):** Percentage of designs violating bounding volume constraints. **Design Space Violation Magnitude ($\mathbf{DQ}_5$):** The volume of material generated outside the allowable bounding box, normalized by design space volume.

Process efficiency. We measure the average number of iterations required to reach a valid, target-compliant design ($\mathbf{PE}_1$). If no valid design is found within 10 iterations, the run is counted as a failure.

Model	$\mathbf{R}_1 \uparrow$	$\mathbf{R}_2 \uparrow$	$\mathbf{R}_3 \uparrow$
Claude Opus 4.5	96.4%	**88.2%**	81.3%
Claude Sonnet 4.5	97.5%	87.6%	84.5%
Gemini 3 Flash	97.0%	84.4%	85.1%
Gemini 3 Pro	**97.5%**	76.3%	**86.8%**

Table 1: Reliability Metrics: Execution Success (R_1) , Meshing Success (R_2) and Simulation Success (R_3).

We compare the different models and isolate the impact of physics-based feedback on reliability and design quality by performing statistical analyses.

6.2 Implementation Details

We used commercial cloud providers to run the LLMs, a machine with 48GB of VRAM to run the HXT meshing algorithm with `tet4` elements and FEA, and the langgraph orchestration framework is executed on consumer hardware. We set the temperature to 0.5 to promote diverse results, allow for 4096 output tokens and disable thinking. On average a single design interation consumes 12,767 input tokens and 5,606 output tokens that, at the time of writing, costs approximately 0.023 USD (Gemini 3 Flash), 0.093 USD (Gemini 3 Pro), 0.204 USD (Claude Opus 4.5), or 0.122 USD (Claude Sonnet 4.5) in inference costs. The average wall-clock time of one iteration is 28.7±36.9 seconds.

7 Results

The quantitative performance of the different models is summarized in Table 1 for reliability metrics and Table 2 for design quality metrics. Additionally, Table 3 presents the process efficiency in terms of iterations to convergence. Inter-

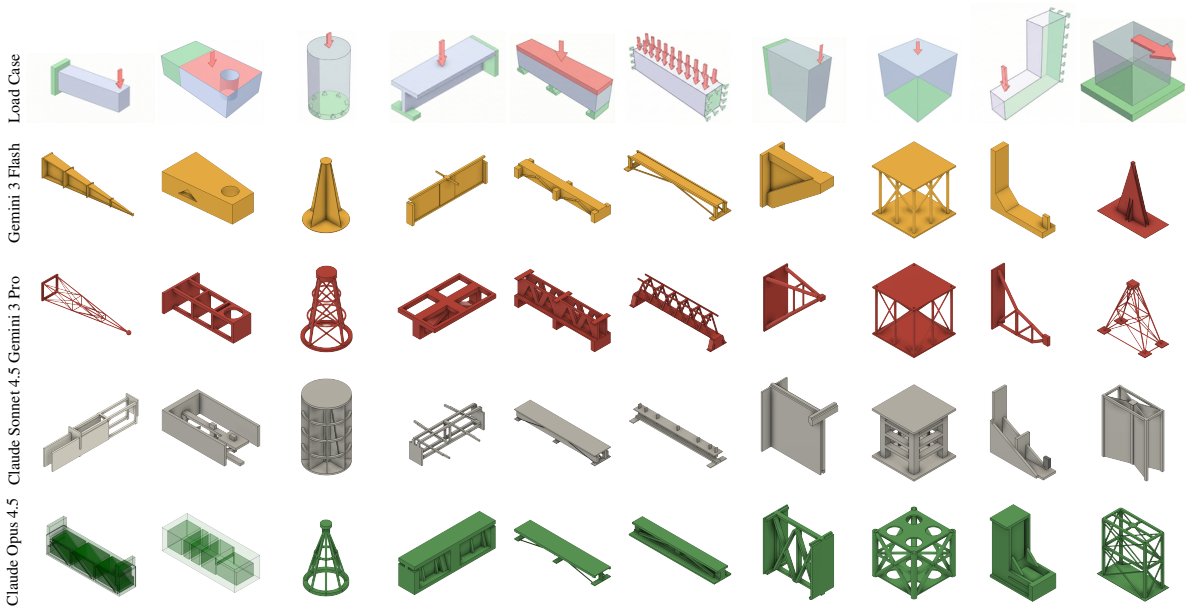

Figure 4: Generated CAD objects of the Hybrid Agentic Architecture. The top row shows visualizations of the input load case (red: forces, green: fixed supports, gray: design space). Subsequent rows show results from Gemini 3 Flash, Gemini 3 Pro, Claude Sonnet 4.5 , and Claude Opus 4.5.

Model	DQ_1	$DQ_2 \uparrow$	DQ_3	$DQ_4 \downarrow$	$DQ_5 \downarrow$
Claude Opus 4.5	4.97±1.6	37.2	85.0	27.7%	0.3%±1.0
Claude Sonnet 4.5	4.96±1.5	14.5	99.7	85.7%	39.6%±64.5
Gemini 3 Flash	4.04±1.5	22.7	63.1	**8.7%**	**0.1%**±0.3
Gemini 3 Pro	4.18±1.5	24.3	120.1	32.3%	12.7%±46.1

Table 2: Design Quality Metrics: Safety Factor (DQ_1), Efficiency (DQ_2) , Geometric Complexity (DQ_3) and Constraint Compliance (DQ_4, DQ_5).

Model	PE_1 Enabled ↓	PE_1 Disabled ↓
Claude Opus 4.5	6.0	10.4
Claude Sonnet 4.5	4.5	13.0
Gemini 3 Flash	3.3	6.3
Gemini 3 Pro	**2.5**	**4.0**

Table 3: Average Iterations to convergence (PE_1) with our agentic framework (Enabled) against single LLM (Disabled).

model comparisons revealed highly significant differences in code execution (H=163.04, p<0.01) and FEA success rates (H=46.24, p<0.01), with Gemini 3 Flash significantly outperforming all models (95.6% execution, 84.4% FEA; all pairwise p<0.01).

8 Ablation

In a series of ablation studies, we dissect the contributions of the physics-in-the-loop feedback mechanism and individual agent roles within the architecture. These experiments aim to isolate the impact of each component on overall system performance, providing insights into the efficacy of the hybrid design approach.

	FEA Enabled			FEA Disabled		
Model	Target ↑	Under ↓	Over ↓	Target ↑	Under ↓	Over ↓
Claude Opus 4.5	**72.7%**	9.1%	**18.2%**	14.3%	28.6%	57.1%
Claude Sonnet 4.5	56.3%	25.0%	18.8%	25.0%	25.0%	50.0%
Gemini 3 Flash	68.2%	**0.0%**	31.8%	20.0%	**0.0%**	80.0%
Gemini 3 Pro	66.7%	**0.0%**	33.3%	**33.3%**	16.7%	**50.0%**

Table 4: Comparison of CAD designs falling within the target safety factor range, underbuilt (< 2) and overbuilt (> 5) with FEA feedback enabled vs. disabled.

8.1 Physics Feedback

We argue that embedding physics-based feedback from FEA into the design loop leads to a marked improvement in the functional validity of generated CAD models. To this end, we conduct an ablation study where we disable the physics-based tools (marked with 🔧 in Figure 3) of the Reviewer Agents, allowing only feedback based on the visual appearance using VLMs. We observe the achieved safety factor (DQ_1) to assess the impact of physics feedback. We set a safety factor target range $[2, 5]$ consistent for general-purpose mechanical engineering [Budynas and Nisbett, 2020], and hypothesize that without physics feedback from FEA, designs will less consistently fall within this range (see Table 4). We performed a Fisher Exact Test [Fisher, 1922] comparing the success rates of achieving target safety factors between FEA-enabled (49/83 successes, 59.0%) and FEA-disabled (6/27 successes, 22.2%) conditions across all models, confirming statistical significance ($p = 0.0008$).

8.2 Iterative Refinement

To isolate the effect of iterative refinement driven by physics feedback, we compare the performance metrics at each iteration of the design loop. Figure 6 illustrates the evolution

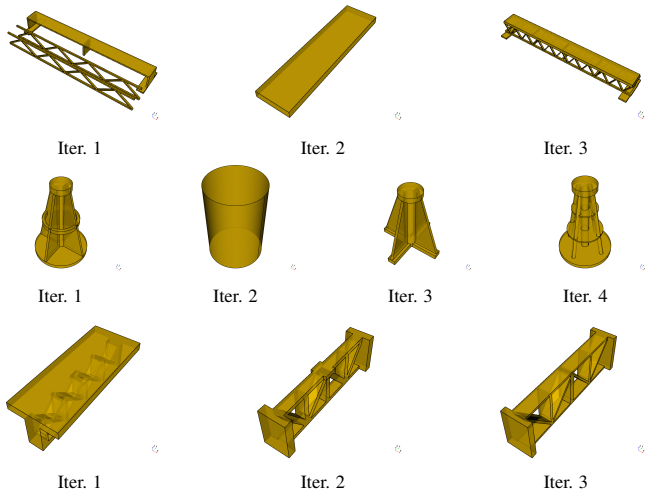

Figure 5: Each row illustrates the iterative refinement of CAD models from initial agent-generated drafts to final physics-validated designs guided by FEA feedback.

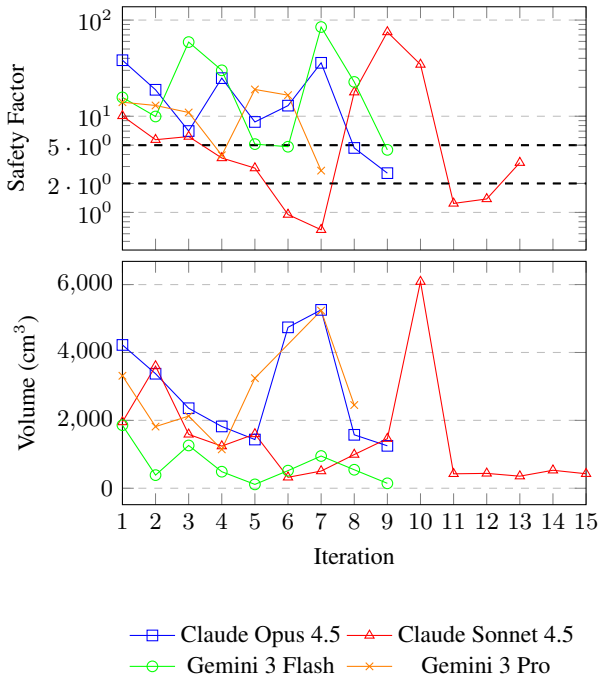

Figure 6: Safety Factor (target range 2–5 dashed, top) and volume (bottom) convergence. Physics feedback guides agents to converge on valid safety factors while progressively reducing volume.

of the safety factor across iterations. The results show that the safety factor initially exceeds the target range but gradually improves with each iteration, converging to an acceptable level after several rounds of refinement. This underscores the efficacy of the physics-in-the-loop feedback mechanism in guiding the design process towards structurally sound solutions.

8.3 Multi-Agent

Next, we aim to isolate the impact of a multi-agent setup vs. a single LLM and remove the planning and review agents. Instead we formulate the task with a single CAD Engineer agent with access to the physics-based tools that directly generates CAD code from the load case description and refines output from raw FEA results. We observe the average safety factor (DQ_1) and convergence (PE_1) for randomly sampled inputs in Table 3 to assess if the planner reduces the load of the engineer, accelerating convergence and if the reviewer's structured feedback improves safety factor. We performed a Welch t-test comparing iterations required with planning enabled ($n = 80$, $M = 4.44$, $SD = 5.36$) versus disabled ($n = 80$, $M = 10.77$, $SD = 10.11$), confirming a significant increase in required iterations without planning ($t = -3.17$, $df = 13.39$, $p = 0.0071$, Cohen's $d = -1.03$).

9 Discussion

We observe that our system is capable of correctly interpreting given load cases and generate complex CAD designs. Visual analysis of the generated CAD code indicates that Claude Sonnet 4.5 frequently produces overly complex geometries that are difficult to mesh and simulate, resulting in lower FEA success rates and high design space violations. In contrast, Gemini 3 Pro generates more efficient designs that balance structural complexity with manufacturability. Notably, the smaller Gemini 3 Flash outperforms Claude Sonnet 4.5 despite its reduced model size. This reveals a "Goldilocks effect" where the mid-tier Gemini 3 Flash excels, likely because it generates simpler, more practical geometries that are easier to mesh and simulate, whereas larger LLMs over-engineer, resulting in modelling errors.

Inspecting the average safety factors in Table 2 of the approved designs, we observe that all models tend to over-engineer, producing safety factors close to the upper limit of the target range (5.0). Opus 4.5 hit the iteration limit most frequently (16.3%), followed by Sonnet 4.5 (12.8%).

9.1 Planning the CAD Design

We see that the Gemini models deteriorate less from the removal of the Planner Agent (see results in Table 3). This suggests that these models possess sufficient reasoning capabilities to decompose the design task internally. However, the Planner Agent still provides value by structuring the CAD Engineer's workflow, leading to more consistent code generation. Without the Planner Agent Claude Sonnet 4.5 and Claude Opus 4.5 exhibit larger degradation, indicating that explicit task decomposition is more important for their success.

9.2 Physics-in-the-Loop

The ablation studies indicate that embedding physics feedback in the design loop enables the system to more effectively correct over- and under-engineered solutions. The FEA validation provides concrete, quantitative performance measures that help guide the CAD Engineer in making targeted design adjustments. This feedback loop enables the system to iteratively refine designs towards optimal structural performance,

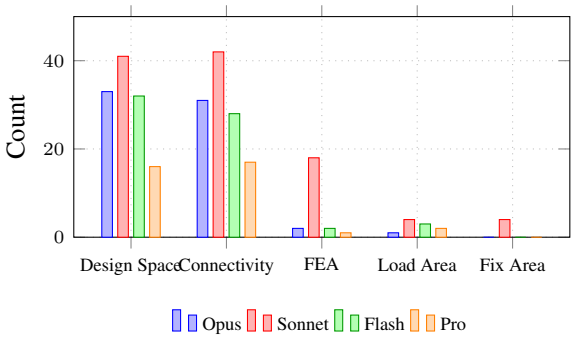

Figure 7: Distribution of failure types by model that can occur due to lacking comprehension of the design task or modelling errors.

reducing reliance on geometric heuristics alone. However, with more iterations, the results tend to become more unstable. Future work could focus on detecting stopping criteria when a CAD design becomes unrecoverable.

9.3 Failure Type Analysis

Designs most commonly fail due to design space violations and disconnected parts. The distribution is consistent across all tested LLMs with Claude Sonnet-4.5 performing worst across of all failure cases. We attribute high design space violations and connectivity errors to the limited spatial reasoning for load paths of LLMs. FEA-related failures are relatively rare, suggesting that when geometries pass meshing, they are generally well-posed for simulation. Rarely it happens that the area where loads or fixed supports are applied is not filled with material. Figure 7 summarizes the failure type distribution.

9.4 Comparison to Other Methods

Since our method is novel in its problem formulation, there are no directly comparable prior works. However, compared to purely generative CAD systems that lack physics validation, our approach demonstrates 3.4% higher compile rate (IR_1) compared to Alrashedy *et al.* [2025]. We can also compare the complexity of generated geometries: our method achieves an average of 83.1 faces (DQ_3) compared to 24.2 faces present in methods based on the DeepCAD dataset [Wu *et al.*, 2021] such as the method by Alrashedy *et al.* and Preintner *et al.*. A qualitative comparison is shown in Figure 8.

9.5 Comparison to Topology Optimization

Topology optimization (TO) methods [Bendsøe and Sigmund, 2013] solves the same problem. We compare the execution time of our method on our dataset with a PyTorch-based TO implementation[4] on a machine with 48GB VRAM, with 10 iterations and same meshing parameters and found that TO takes on average 18.6±1.2 seconds to converge whereas our method requires 28.7±36.9 seconds. However, our approach has advantage that it produces editable CAD models directly with higher fidelity (example shown in Figure 8) while TO produce voxel or mesh-based representations that require additional post-processing like manually tracing design in CAD software.

[4]https://github.com/meyer-nils/torch-fem

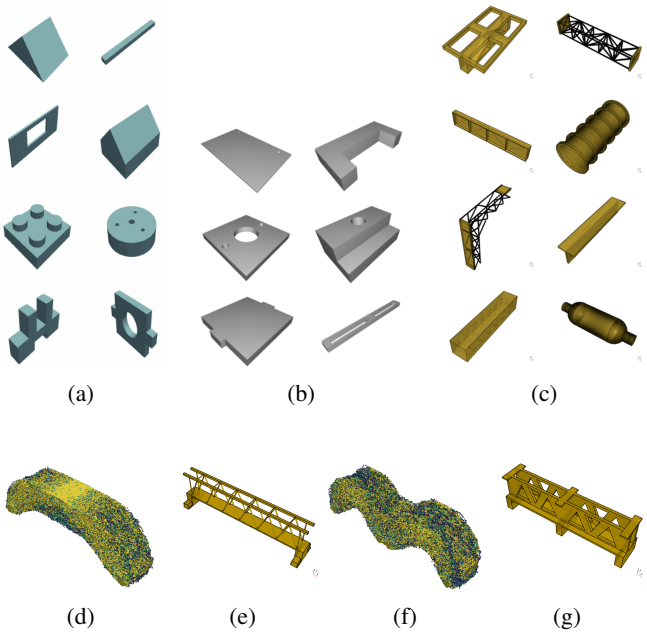

Figure 8: Qualitative comparison showing the fidelity of (a) CadCodeVerify [Alrashedy *et al.*, 2025], (b) EvoCAD [Preintner *et al.*, 2025], (c) our method. We compare outputs of topology optimization (d, f), and our method (e, g) the same load case with our method.

10 Conclusion

Extensions to dynamic or thermal scenarios or optimization for further criteria, such as sustainability, are left for future research. This can include integrating material selection, manufacturability assessments, and cost-analysis into the design loop. Architecturally, the current prompt-driven incorporation of physics feedback could be replaced with reinforcement learning-based optimization.

We present a hybrid agentic-physical system that combines LLM-based agents with mature engineering tools. The proposed approach generates CAD designs based on load bearing requirements that are structurally grounded and more geometrically complex than those generated by geometry optimized methods, by embedding physics-based validation directly into the agents' decision-making loop. Our results suggest that combining agentic reasoning with knowledge-based engineering tools offers a practical foundation for trustworthy AI-assisted systems in engineering design.

References

[Alam and Ahmed, 2025] Md Ferdous Alam and Faez Ahmed. GenCAD: Image-Conditioned Computer-Aided Design Generation with Transformer-Based Contrastive Representation and Diffusion Priors. *http://arxiv.org/abs/2409.16294*, 2025.

[Alrashedy *et al.*, 2025] Kamel Alrashedy, Pradyumna Tambwekar, Zulfiqar Zaidi, Megan Langwasser, Wei Xu, and Matthew Gombolay. Generating CAD Code with Vision-Language Models for 3D Designs. 10.48550/arXiv.2410.05340, February 2025.

[Bendsøe and Sigmund, 2013] Martin Philip Bendsøe and Ole Sigmund. *Topology Optimization: Theory, Methods, and Applications*. Springer, Berlin / Heidelberg, 2 edition, 2013.

[Berger *et al.*, 2025] Elias Berger, Maximilian Peter Dammann, Jan Mehlstäubl, Bernhard Saske, Felix Braun, and Kristin Paetzold-Byhain. Challenges and opportunities in the integration of generative AI with computer-aided design. In *Proceedings of the Design Society*, volume 5, pages 881–890, 2025.

[Berger *et al.*, 2026a] Elias Berger, Kevin Herrmann, Felix Pusch, Tobias Kriesell, Paul Gembarski, Jan Mehlstäubl, Roland Lachmayer, and Kristin Paetzold-Byhain. From geometry to function: Towards context-aware generative ai for engineering design. In *Proceedings of the Design Society*, 2026.

[Berger *et al.*, 2026b] Elias Berger, Jan Mehlstäubl, Bernhard Saske, and Kristin Paetzold-Byhain. Multi-task cad generation using compact decoder-only models. *2026 International Conference on Advances in Artificial Intelligence and Machine Learning (AAIML)*, pages 1063–1070, 2026.

[Budynas and Nisbett, 2020] Richard G. Budynas and J. Keith Nisbett. *Shigley's Mechanical Engineering Design*. McGraw-Hill Education, New York, NY, 11 edition, 2020.

[Doris *et al.*, 2025] Anna C. Doris, Md Ferdous Alam, Amin Heyrani Nobari, and Faez Ahmed. CAD-Coder: An Open-Source Vision-Language Model for Computer-Aided Design Code Generation. 10.48550/arXiv.2505.14646, May 2025.

[Dupont *et al.*, 2024] Elona Dupont, Kseniya Cherenkova, Dimitrios Mallis, Gleb Gusev, Anis Kacem, and Djamila Aouada. TransCAD: A hierarchical transformer for CAD sequence inference from point clouds. arXiv:2407.12702, 2024.

[Fisher, 1922] Ronald A. Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94, 1922.

[Govindarajan *et al.*, 2026] Prashant Govindarajan, Davide Baldelli, Jay Pathak, Quentin Fournier, and Sarath Chandar. CADmium: Fine-Tuning Code Language Models for Text-Driven Sequential CAD Design. 10.48550/arXiv.2507.09792, January 2026.

[He *et al.*, 2025] Changqi He, Shuhan Zhang, Liguo Zhang, and Jiajun Miao. CAD-Coder:Text-Guided CAD Files Code Generation. 10.48550/arXiv.2505.08686, May 2025.

[Herrmann *et al.*, 2021] K. Herrmann, O. Altun, P. Wolniak, I. Mozgova, and R. Lachmayer. Methodischer aufbau von entwicklungsumgebungen nach dem generative parametric design approach. In *Proceedings of the 32nd Symposium Design for X, DFX 2021*, 2021.

[Hirz *et al.*, 2011] Mario Hirz, Alexander Harrich, and Peter Rossbacher. Advanced computer aided design methods for integrated virtual product development processes. *Computer-Aided Design and Applications*, 8(6):901–913, 2011.

[Huang, 2024] Xiaocheng Huang. Understanding the planning of LLM agents: A survey. *arXiv preprint*, 2024.

[Jayaraman *et al.*, 2023] Pradeep Kumar Jayaraman, Joseph G. Lambourne, Nishkrit Desai, Karl D. D. Willis, Aditya Sanghi, and Nigel J. W. Morris. Solidgen: An autoregressive model for direct b-rep synthesis. arXiv:2203.13944, 2023.

[Khan *et al.*, 2024a] Mohammad Sadil Khan, Elona Dupont, Sk Aziz Ali, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. CAD-SIGNet: CAD Language Inference from Point Clouds using Layer-wise Sketch Instance Guided Attention. 10.48550/arXiv.2402.17678, February 2024.

[Khan *et al.*, 2024b] Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin Sheikh, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. Text2CAD: Generating Sequential CAD Models from Beginner-to-Expert Level Text Prompts, 2024.

[Kolodiazhnyi *et al.*, 2025] Maksim Kolodiazhnyi, Denis Tarasov, Dmitrii Zhemchuzhnikov, Alexander Nikulin, Ilya Zisman, Anna Vorontsova, Anton Konushin, Vladislav Kurenkov, and Danila Rukhovich. Cadrille: Multi-modal CAD Reconstruction with Online Reinforcement Learning. 10.48550/arXiv.2505.22914, September 2025.

[Lambourne *et al.*, 2021] Joseph G. Lambourne, Karl D. D. Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. Brepnet: A topological message passing system for solid models. arXiv:2104.00706, 2021.

[Lv and Bao, 2025] Chaofan Lv and Jinsong Bao. CADInstruct: A multimodal dataset for natural language-guided CAD program synthesis. *Computer-Aided Design*, 188:103926, 2025.

[Mahamid *et al.*, 2020] Mustafa Mahamid, Edwin H. Gaylord, and Charles N. Gaylord, editors. *Structural Engineering Handbook*. McGraw-Hill, New York, 5th edition, 2020. Comprehensive reference on structural elements such as beams, columns, trusses, slabs, and frames.

[Nash *et al.*, 2020] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. PolyGen: An Autoregressive Generative Model of 3D Meshes. 10.48550/arXiv.2002.10880, February 2020.

[Ocker *et al.*, 2025] Felix Ocker, Stefan Menzel, Ahmed Sadik, and Thiago Rios. From Idea to CAD: A Language Model-Driven Multi-Agent System for Collaborative Design. 10.48550/arXiv.2503.04417, March 2025.

[Pahl *et al.*, 2007] Gerhard Pahl, Wolfgang Beitz, Jörg Feldhusen, and Karl-Heinrich Grote. *Engineering Design: A Systematic Approach*. Springer, 2007.

[Plaat *et al.*, 2025] Aske Plaat, Max Van Duijn, Niki Van Stein, Mike Preuss, Peter Van der Putten, and Kees Joost Batenburg. Agentic large language models, a survey. *Journal of Artificial Intelligence Research*, 84, December 2025.

[Preintner *et al.*, 2025] Tobias Preintner, Weixuan Yuan, Adrian König, Thomas Bäck, Elena Raponi, and Niki van Stein. EvoCAD: Evolutionary CAD code generation with vision language models. arXiv:2510.11631, 2025.

[Regassa Hunde and Debebe Woldeyohannes, 2022] Bonsa Regassa Hunde and Abraham Debebe Woldeyohannes. Future prospects of computer-aided design (cad) – a review from the perspective of artificial intelligence (ai), extended reality, and 3d printing. *Results in Engineering*, 14:100478, 2022.

[Schulte *et al.*, 1993] Michael Schulte, Christian Weber, and Rainer Stark. Functional features for design in mechanical engineering. *Special Issue: CARs & FOF '92*, 23(1):15–24, November 1993.

[Shen, 2024] Zhuocheng Shen. LLM with tools: A survey. *arXiv preprint*, 2024.

[Steininger *et al.*, 2025] Sarah Steininger, Jasmin Zhao, and Johannes Fottner. Enhancing computer-aided design with deep learning frameworks: A literature review. *Proceedings of the Design Society*, 5:1515–1524, August 2025.

[T. Chen *et al.*, 2025] T. Chen, C. Yu, Y. Hu, J. Li, T. Xu, R. Cao, L. Zhu, Y. Zang, Y. Zhang, Z. Li, and L. Sun. Img2CAD: Conditioned 3-D CAD Model Generation From Single Image With Structured Visual Geometry. *IEEE Transactions on Industrial Informatics*, 21(11):8539–8549, November 2025.

[Usama *et al.*, 2025] Muhammad Usama, Mohammad Sadil Khan, Didier Stricker, and Muhammad Zeshan Afzal. NURBGen: High-fidelity text-to-CAD generation through LLM-driven NURBS modeling. arXiv:2511.06194, 2025.

[Wang *et al.*, 2025a] Ruiyu Wang, Yu Yuan, Shizhao Sun, and Jiang Bian. Text-to-CAD Generation Through Infusing Visual Feedback in Large Language Models. 10.48550/arXiv.2501.19054, June 2025.

[Wang *et al.*, 2025b] Siyu Wang, Cailian Chen, Xinyi Le, Qimin Xu, Lei Xu, Yanzhou Zhang, and Jie Yang. CAD-GPT: Synthesising CAD Construction Sequence with Spatial Reasoning-Enhanced Multimodal LLMs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(8):7880–7888, April 2025.

[Willis *et al.*, 2021] Karl D. D. Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G. Lambourne, Armando Solar-Lezama, and Wojciech Matusik. Fusion 360 Gallery: A Dataset and Environment for Programmatic CAD Construction from Human Design Sequences. 10.48550/arXiv.2010.02392, May 2021.

[Wu *et al.*, 2021] Rundi Wu, Chang Xiao, and Changxi Zheng. DeepCAD: A Deep Generative Network for Computer-Aided Design Models. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6752–6762. IEEE, 2021.

[Xu *et al.*, 2025] Jingwei Xu, Zibo Zhao, Chenyu Wang, Wen Liu, Yi Ma, and Shenghua Gao. CAD-MLLM: Unifying Multimodality-Conditioned CAD Generation With MLLM. 10.48550/arXiv.2411.04954, March 2025.

[Yao *et al.*, 2023] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint*, 2023.

[Zhang *et al.*, 2025] Wei Zhang, Wei Chen, and Liang Zhao. A survey on generative AI for CAD and mechanical design. *Computer-Aided Design*, 2025.

[Zhou *et al.*, 2023] Shengdi Zhou, Tianyi Tang, and Bin Zhou. CADParser: A Learning Approach of Sequence Modeling for B-Rep CAD. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 1804–1812, Macau, SAR China, August 2023. International Joint Conferences on Artificial Intelligence Organization.