realised.io

Interactive AI-Assisted CAD: Better than either alone

1. Introduction

Computer Aided Design (CAD) systems are central to modern engineering, enabling the creation of precise, parametric models that can be analysed, manufactured, and refined interactively. Despite their sophistication, applications such as Autodesk Fusion 360 and FreeCAD remain difficult to use effectively, requiring spatial reasoning, technical expertise, and familiarity with complex workflows. This creates a divide between those who can imagine product ideas and those able to translate them into functional CAD models.

Recent advances in AI offer a potential ways to bridge this gap. These systems excel at interpreting natural language and generating structured outputs such as code, driving growing interest in AI assisted design automation. In principle, users could describe objects in natural language and have AI generate corresponding CAD models, lowering the barrier to design and prototyping. In practice, however, major limitations remain. CAD modelling is not simply generative but constraint-driven, requiring geometric validity, feature consistency, and adherence to physical requirements. LLMs, which rely on probabilistic token prediction, lack a true understanding of geometry or spatial constraints. As a result, they often generate outputs that appear plausible but fail in real CAD environments through incorrect references, inconsistent dimensions, or invalid geometry.

This report argues that effective AI-assisted CAD systems must move beyond one shot generation toward iterative, human-guided workflows. In this approach, LLMs act not as autonomous designers but as collaborators that generate executable CAD scripts. These scripts are validated by deterministic CAD engines and refined through human feedback, which provides spatial intuition, design intent, and contextual judgement beyond current AI capabilities. The workflow therefore becomes one of augmentation rather than replacement, combining human reasoning with machine generation. Once a validated, modular CAD model exists, AI can extend beyond geometry creation and support broader product lifecycle tasks.

To develop these ideas, this report draws on observations from the development of my ElixiHub application, an AI-driven CAD orchestration system combining multi-agent generation, FreeCAD-based validation, and human-guided refinement. While not the primary focus, ElixiHub highlights the challenges of integrating LLMs with CAD systems and demonstrates the importance of structured workflows, component decomposition, and feedback-driven iteration.

2. Background and Problem Setting

2.1 The Nature of CAD: Deterministic, Constraint-Driven Design

At its core, CAD modelling is a deterministic process governed by geometric rules, constraints, and feature dependencies. Parametric CAD systems define objects not as static shapes but as sequences of operations—extrusions, cuts, fillets, and assemblies—parameterised by dimensions and relationships. This allows designs to be modified and regenerated reliably, making parametric modelling essential for engineering and manufacturing workflows.

However, this structure introduces significant complexity. Each feature depends on prior geometry, and errors can propagate through the model if dependencies are not valid. CAD resolver engines enforce strict validity requirements, meaning that even small inconsistencies—such as referencing a non-existent edge or creating overlapping geometry—can cause a model to fail entirely. As a result, successful CAD modelling requires not only geometric understanding but also an awareness of feature sequencing, constraint logic and system behaviour.

This deterministic and tightly constrained environment contrasts sharply with the flexible, approximate nature of most AI-generated outputs.

2.2 Generative AI and the limitations of its Design Capability

Large language models have demonstrated remarkable ability in generating structured outputs, including code. Tools such as CADQuery allow CAD models to be expressed programmatically through Python scripts, creating a potential bridge between natural language input and parametric geometry. Recent research has explored this direction further. For example, [1]Ruiyu Wang et al. (2025) proposed a text-to-CAD generation framework that incorporates visual feedback into large language models, highlighting the importance of iterative validation and refinement in CAD generation workflows.

Initially, this approach appears viable because LLMs can generate syntactically correct scripts capable of producing valid CAD models [2]. However, these models are often brittle. While the generated code may appear plausible, the resulting geometry frequently fails to match user expectations except in relatively trivial designs.

2.3 Fundamental Limitations of LLMs in CAD Contexts

The limitations of LLMs in CAD are structural rather than incidental. LLMs do not possess an internal model of 3D space. They cannot reliably reason about:

- Relative positioning of features

- Intersections and clearances
- Structural relationships between components

As a result, they often produce designs that are geometrically invalid or physically impractical.

CAD modelling requires satisfying explicit and implicit constraints:

- Dimensions must be consistent
- Features must align correctly
- Assemblies must respect interfaces

LLMs do not inherently track or enforce these constraints. They generate sequences of tokens without maintaining a persistent understanding of constraint satisfaction, leading to frequent inconsistencies.

Sensitivity to Minor Errors

In many AI domains, small errors are tolerable (for example image generation). A single incorrect reference or dimension in CAD can cause:

- Script execution failure
- Broken model topology
- Downstream assembly issues

This brittleness makes one-shot AI generation practically difficult to rely on.

2.4 Human–AI Collaboration as a Necessary Paradigm

Given these limitations, fully autonomous AI-driven CAD generation is both unrealistic and undesirable. Instead we must leverage the complementary strengths of humans and AI.

Humans provide:

- The motivation and direction to produce the project
- Spatial intuition
- Functional understanding of objects
- Ability to recognise invalid or impractical designs
- High-level design intent and constraints

AI provides:

- Code synthesis and modification
- Exploration of design variations
- A wealth of information related to materials and construction

The interaction between these roles is critical. Humans guide the process by refining prompts, correcting errors, and validating outputs, while AI accelerates iteration by generating and adapting CAD scripts. This collaborative loop allows the system to converge on viable designs that neither human nor AI could efficiently produce alone.

2.5 Identified Gap

Existing AI-assisted design approaches largely focus on generating geometry, either as meshes or scripts, without addressing:

- Validation
- Iterative correction
- Human collaboration

This leaves a critical gap between AI-generated outputs and usable engineering artefacts. The approach explored in this report addresses this gap through an **iterative AI–CAD feedback loop**, combined with human guidance and component-based design. This framework provides a pathway toward reliable AI-assisted CAD while unlocking broader applications across the product development pipeline.

3. Proposed Approach: Iterative AI–CAD Feedback Loop

3.1 Overview

To address the limitations of large language models in CAD contexts, this report proposes an **iterative AI CAD feedback loop** in which generative models, deterministic CAD engines, and human users operate as a coordinated system. Rather than attempting to generate complete and correct models in a single step, the approach decomposes the design process into a sequence of executable iterations, each informed by validation feedback and human input. At a high level, the workflow consists of the following stages:

1. A user provides a natural-language design brief, optionally supported by reference images
2. An LLM breaks down the project into components and generates a candidate parametric CAD scripts
3. The scripts are executed within a CAD engine
4. The resulting geometries are created and any errors are returned to the agent
5. The AI refines the script based on feedback
6. The user reviews and guides further iteration

This loop continues until a valid set of components are created and then an assembly of the components are created. The user reviews to see how well they components fit together.

3.2 Coding Agents as the Interface Layer

The key design decision in this approach is using code as the interface between AI and CAD systems. Rather than generating geometry directly, the LLM produces scripts that define parametric models using libraries. The code is:

- **Structured and explicit.** Operations such as sketches, extrusions, and fillets are clearly defined, reducing ambiguity and providing a traceable design process.
- **Executable.** Scripts can run inside a CAD environment, enabling immediate validation of geometric correctness. This shifts the problem from static generation to a generate–execute–evaluate workflow.
- **Editable and iterative.** Features can be adjusted incrementally, errors corrected locally, and designs refined over time. This aligns naturally with both human workflows and the strengths of LLMs, which are effective at modifying existing code from feedback.

By operating at the code level, the system bridges probabilistic AI generation with deterministic CAD execution.

3.3 The Role of the CAD Engine as Ground Truth

In this framework, the CAD engine plays a critical role as the **source of truth** for geometric validity. Systems such as FreeCAD provide a robust kernel capable of enforcing constraints, resolving feature dependencies, and detecting invalid operations.

When a generated script is executed, the CAD engine performs several functions:

- Validates geometric operations
- Enforces dimensional and topological consistency
- Produces a concrete model (e.g. STEP or STL)
- Returns detailed error information if execution fails

This shifts responsibility for correctness away from the LLM and into a system specifically designed for that purpose. Importantly, it also provides **structured feedback** that can be used to guide subsequent iterations.

For example, if a script fails due to referencing a non-existent edge or applying a fillet to incompatible geometry, the error message becomes an input to the next generation step. In this way, the CAD engine enables a feedback loop that progressively constrains the solution space toward valid designs.

3.4 Iteration as part of the design mechanism

A defining characteristic of this approach is the elevation of iteration from a fallback mechanism to a core design principle.

In traditional generative AI workflows, a prompt is issued and a single output is produced. In contrast, CAD modelling is inherently iterative. Even experienced designers refine their models through successive adjustments, responding to both visual inspection and constraint violations.

The proposed system mirrors this process:

- Initial outputs are treated as drafts rather than final solutions
- Errors are expected and incorporated into the workflow
- Each iteration moves the design closer to validity and intent

This iterative loop addresses several of the limitations outlined earlier. It compensates for the lack of internal constraint tracking in LLMs by externalising constraint enforcement to the CAD engine. It also allows the system to recover from errors, rather than failing outright.

Crucially, iteration enables convergence. While any individual generation may be flawed, repeated cycles of execution and correction can produce reliable results.

3.5 Component-Based Decomposition

A further extension of the approach is the decomposition of designs into **multiple interacting components** rather than treating the model as a single monolithic entity.

Real-world products typically consist of:

- Separate parts
- Defined interfaces
- Assembly relationships

By structuring the problem in this way, the system can:

- Generate components independently
- Manage dependencies between parts
- Allow partial regeneration without affecting the entire model

Each component can be treated as its own iterative loop, with its own script, validation cycle, and refinement process. This reduces complexity and improves reliability, as errors are localised rather than propagating across the entire design.

It also aligns closely with how CAD systems and engineering workflows are structured, making the approach more practical for real-world use.

3.6 From Geometry to Artefacts

An important outcome of this workflow is that it produces not just geometry, but usable artefacts. Once a model is successfully generated and validated, it can be exported as:

- STEP (for parametric interoperability)
- STL (for manufacturing, e.g. 3D printing)
- FCStd files that are editable in FreeCAD

Because the model is defined through code, it also retains its parametric structure, allowing further modification and reuse. This distinguishes the approach from mesh-based generation methods, which often produce static outputs with limited editability.

---

4. Case Study: Bed Frame via ElixiHub

4.1 Case Study: Multi-Component Bed Frame Generated via ElixiHub

To evaluate the proposed AI CAD workflow in a realistic setting, a multi-component furniture design task was executed using ElixiHub an application I have developed. The objective was to generate a contemporary wooden king-size platform bed frame based on the following high-level prompt:

“Design a simple contemporary wooden king size platform bed frame with rectangular headboard, lower footboard, side rails, internal support beams, and evenly spaced wooden slats (furniture) — 2170 × 2070 × 1100 mm”

This example represents a significantly more demanding task than simple geometric modelling. The design requires:

- Multiple structurally dependent components
- Precise spatial relationships between parts
- Functional joinery (mortise and tenon connections)
- Assembly-level coherence

Unlike trivial CAD generation tasks, this problem involves both **geometric correctness and functional design intent**, making it a suitable test of the proposed system.

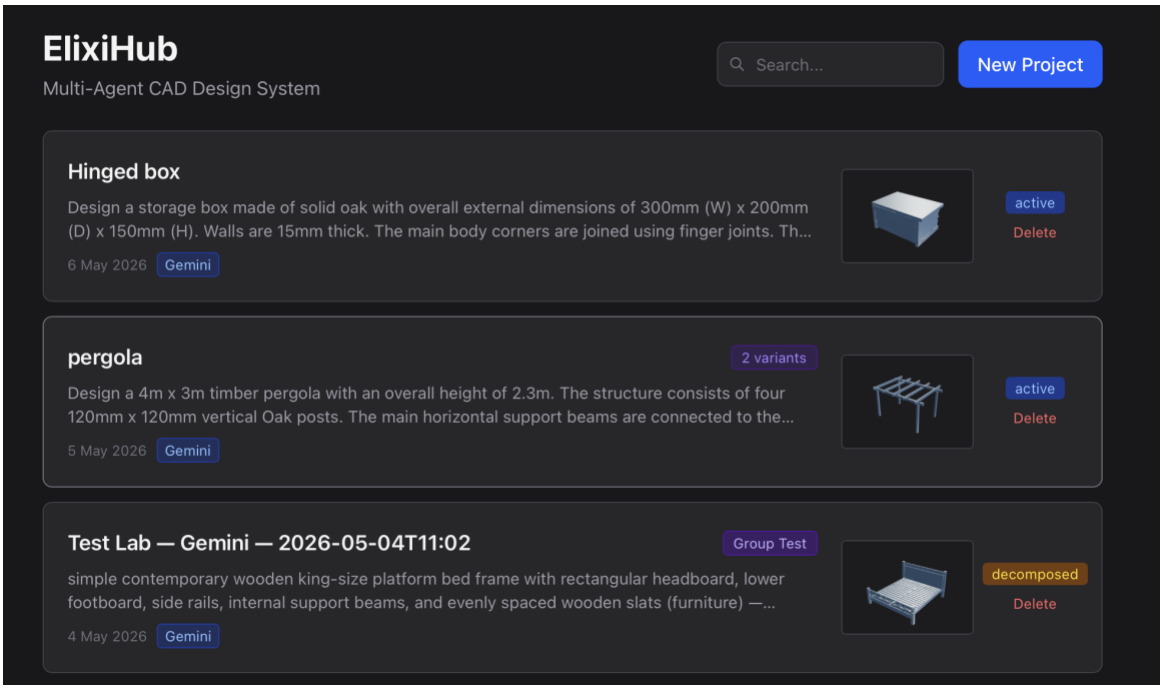

Figure1 : ElixiHub project listings

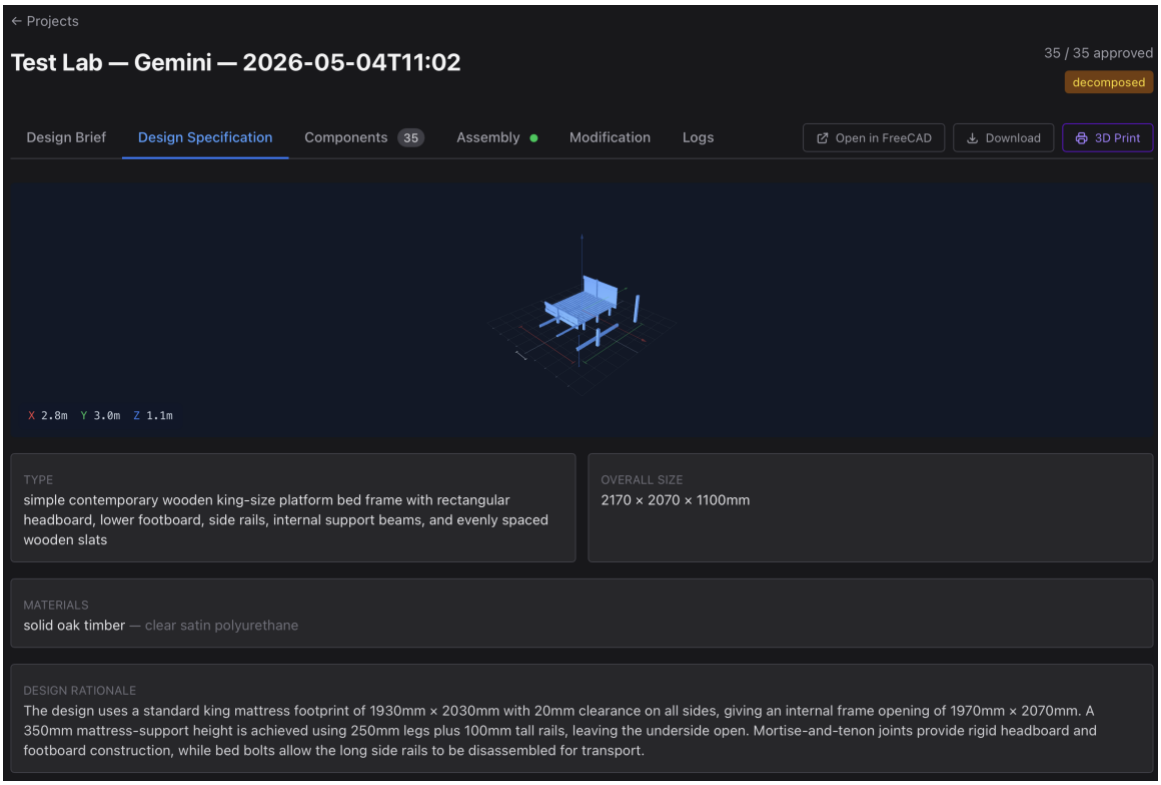

Figure 2 : ElixiHub bed project design specification

4.2 Iterative Human–AI Interaction

A key feature of this case study is the interactive, iterative dialogue between the user and the system, mediated by a large language model (in this instance, Google Gemini).

Rather than producing a complete design in a single step, the system engaged in a back-and-forth process in which. Through iterative prompt refinement and component decomposition, the system converged on a valid multi-part design.

This interaction highlights a critical limitation of LLMs: **they do not reliably infer complete design intent from a single prompt**. Important aspects such as proportions, joinery details, and structural relationships often require clarification or refinement.

Human input in this loop provided:

- Clarification of ambiguous requirements
- Validation of proportions and layout
- Direction for functional features (e.g. joinery)

This reinforces the idea that **human spatial reasoning is essential for guiding the design toward a viable solution**, particularly in complex, multi-part systems.

4.3 Component-Based Generation

The bed frame was decomposed into a set of distinct components, including:

- Headboard
- Footboard
- Internal support beams
- Wooden slats

Each component was treated as an independent unit of generation within ElixiHub, with its own lifecycle, script generation process, and validation loop.

This decomposition proved critical for success. Attempting to generate the entire bed as a single model would require the LLM to maintain consistency across multiple interacting parts—something it struggles to do reliably. Component level generation allowed for:

- Localised reasoning about geometry
- Independent error correction
- Clear separation of responsibilities

This aligns with a broader insight: **LLMs perform significantly better when complex design problems are broken into smaller, well-defined sub-problems**.

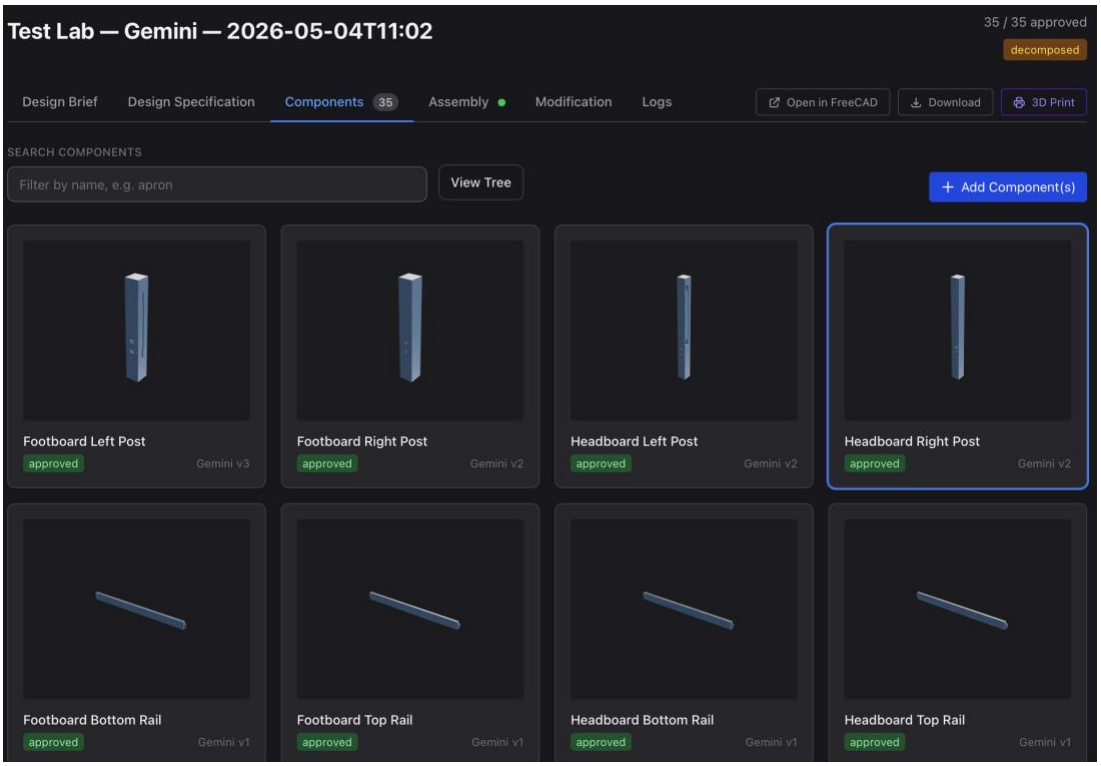

Figure 3 : ElixiHub bed project components view

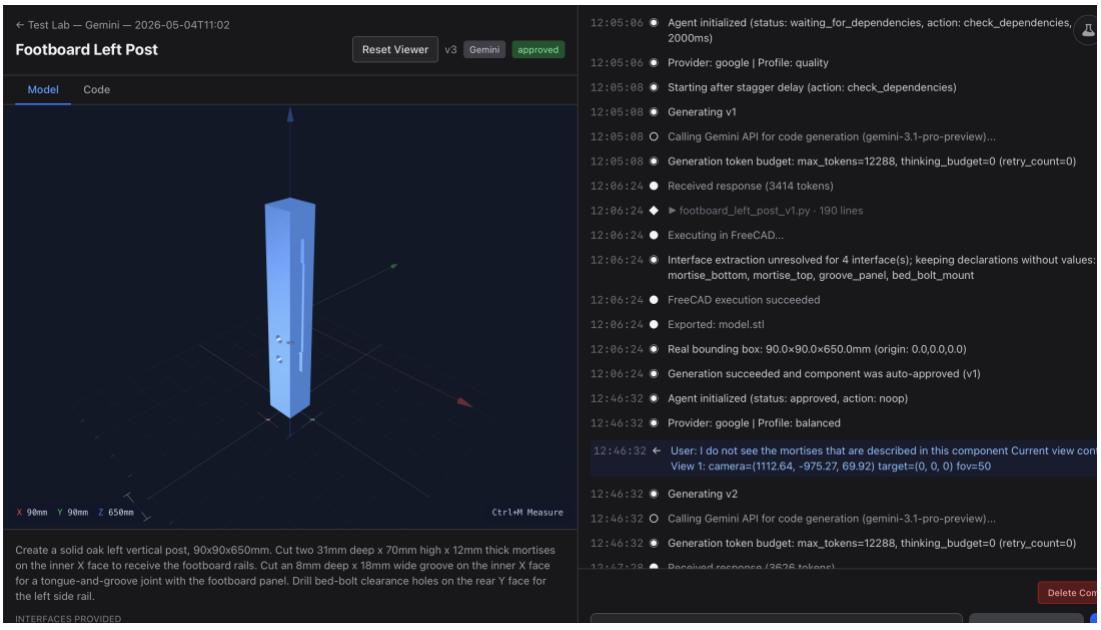

Figure 4 : ElixiHub individual component view

4.4 Dependency Management and Assembly Coherence

While components were generated independently, the overall design required strict coordination between them. For example:

- Side rails needed to align precisely with the headboard and footboard
- Support beams required correct positioning relative to the frame
- Slats needed to be evenly spaced and properly supported

ElixiHub addressed this through a dependency-aware system in which components could:

- Define geometric interfaces (e.g. connection points, dimensions)
- Wait for upstream components before generating dependent geometry
- Regenerate when dependencies changed

This ensured that the final assembly was coherent and structurally consistent.

Importantly, this coordination was not handled implicitly by the LLM. Instead, it was enforced through the system architecture, again highlighting a limitation of current models: they do not naturally maintain consistent relationships across multiple generated artefacts without external structure.

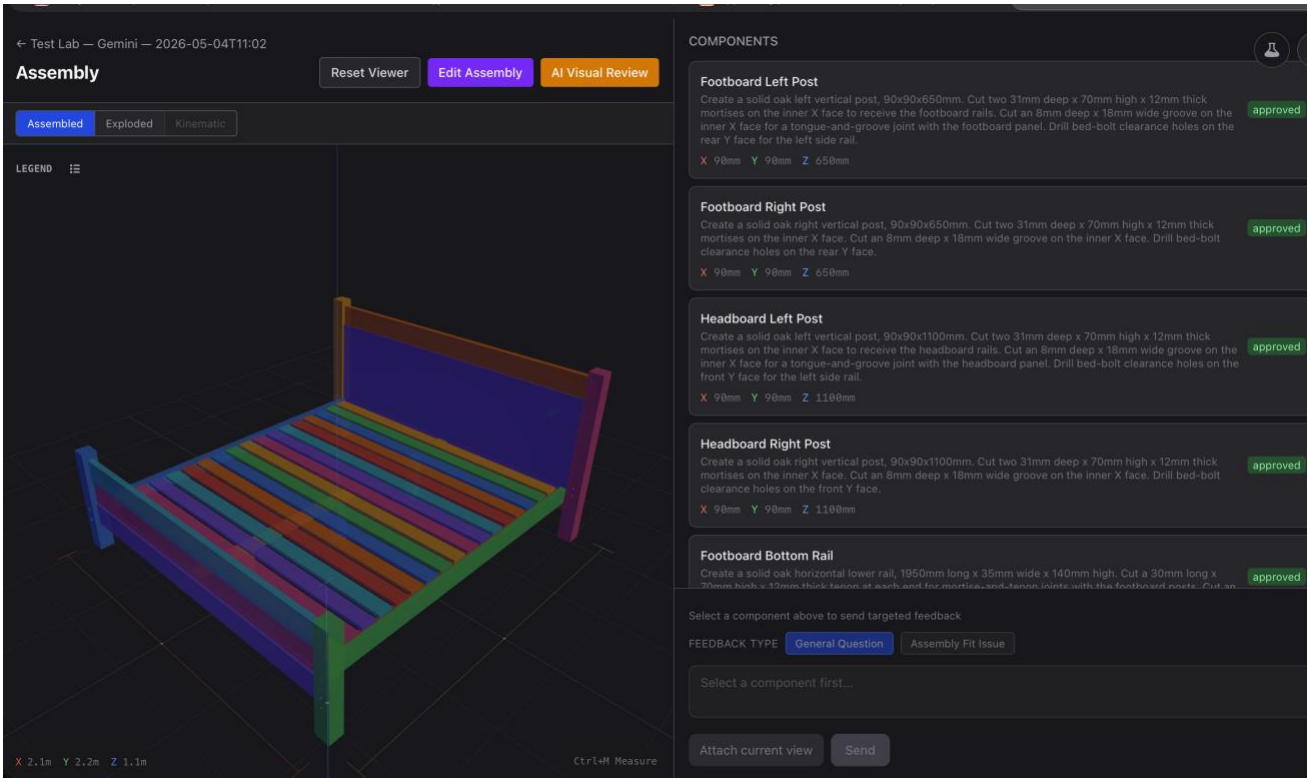

Figure 5: ElixiHub assembly view

4.5 Emergence of Valid Joinery

One of the most notable outcomes of this case study was the successful generation of **mortise and tenon joinery** within the model.

Joinery introduces an additional layer of complexity beyond basic geometry:

- Precise alignment between mating parts
- Consideration of tolerances and fit
- Structural integrity of connections

The fact that valid joinery was produced demonstrates the potential of the iterative approach. However, it is important to note that this was not the result of a single generation step. Instead, it emerged through:

- Iterative refinement of component geometry
- Feedback from CAD execution
- Human validation and guidance

This reinforces a key argument: **LLMs can contribute to sophisticated design features, but only within a system that supports iteration, validation, and guided correction**.

4.6 Error Correction and Convergence

As with other experiments, initial generations frequently failed. Common issues included:

- Misaligned components
- Invalid feature references
- Incomplete geometry definitions

ElixiHub's automatic correction loop played a central role in resolving these issues. By feeding execution errors back into the model, the system was able to progressively refine scripts until valid outputs were achieved.

4.7 Human Validation and Practical Design Judgement

While the CAD engine ensured geometric validity, it did not guarantee that the design was practical or desirable. Human involvement was therefore essential in:

- Assessing proportions and aesthetics
- Evaluating usability and ergonomics

- Ensuring structural plausibility
- Approving final components

For example, while a CAD model might technically assemble correctly, it may still be unsuitable due to poor proportions or impractical construction details. These considerations require **contextual and experiential judgement** that current AI systems cannot reliably replicate.

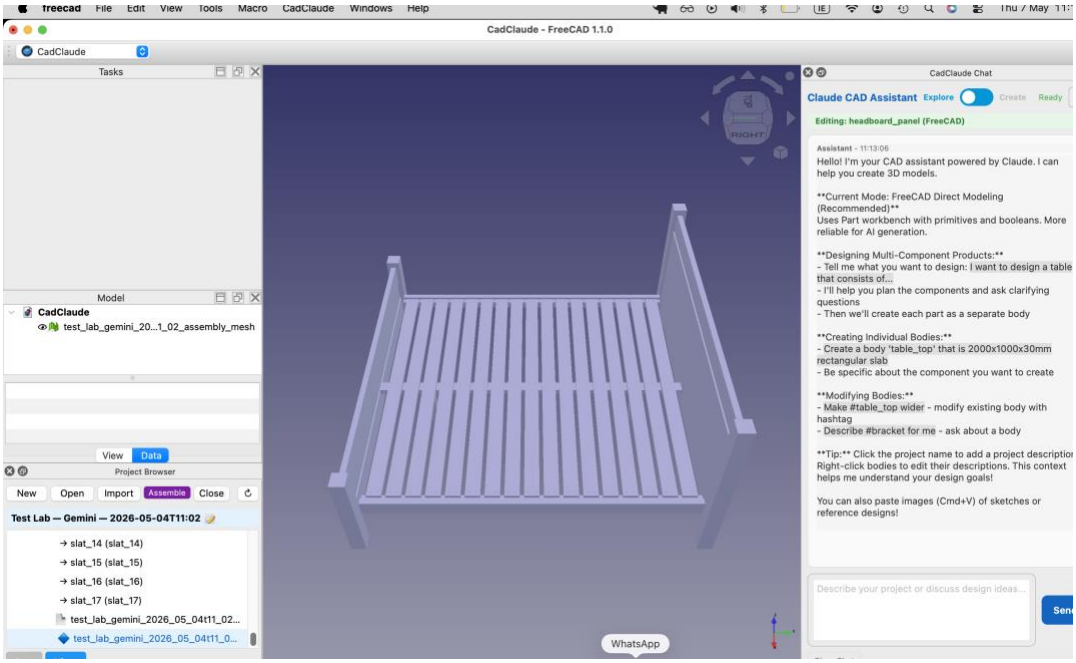

Figure 6: Bed project opened in FreeCAD where refined changes can be applied

---

4.8 From Model to Production

Once the bed frame model was successfully generated and assembled, it provided a structured foundation for further product enhancement tasks. Because the system maintained a component based representation, it becomes possible to use AI to assist with:

Manufacturing Planning

- Suggested fabrication processes
- Assembly sequencing/instructions
- Cost calculation

Structural Analysis (FEA Preparation)

- Identification of load bearing components
- Suggestion of boundary conditions for analysis
- Highlighting potential stress points

Rendering and Presentation

- Generation of visualisations
- Creation of product descriptions
- Documentation for end users or manufacturers

This demonstrates a broader insight: **once a valid, structured CAD model exists, AI can provide significant value across the entire product lifecycle**, not just in initial design generation.

4.9 Key Insights

This case study provides strong empirical support for the proposed approach. The following insights emerge:

- LLMs alone cannot reliably generate complex CAD models
- Iterative feedback and execution-based validation are essential
- Human spatial reasoning is critical for guiding design intent
- Component-based decomposition significantly improves reliability
- System level coordination is required to maintain assembly coherence

---

5. ElixiHub System Insights

5.1 Reframing AI in CAD: From Generation to Systems Integration

Current discussions around generative AI focuses on the idea of one-shot generation, where a user provides a prompt and receives a complete output. This paradigm is appears to be fundamentally incompatible with CAD, where correctness is dictated by strict geometric and functional constraints. The findings of this report suggest that AI in CAD should instead be understood as part of a systems level integration problem. The effectiveness of the approach does not depend on improvements in model capability alone but from the interaction between:

- Generative AI (for proposing candidate solutions)
- Deterministic CAD engines (for enforcing correctness)
- Human users (for providing intent and spatial reasoning)

This structure shifts the focus from what the model can generate to how the system converges on a valid design. In this context, iteration, validation, and coordination become more important than raw generative ability.

5.2 Strengths of the Proposed Approach

Robustness Through Validation

By externalising correctness to a CAD engine such as FreeCAD, the system avoids relying on the LLM to produce valid geometry. This significantly improves reliability, as invalid outputs are detected and corrected rather than silently propagated.

Convergence Through Iteration

The use of iterative refinement allows the system to recover from errors and progressively improve outputs. This is particularly important given the probabilistic nature of LLMs, where initial generations are often incorrect.

Scalability via Decomposition

Component-based design enables complex systems to be broken into manageable units. This reduces cognitive and computational complexity, allowing both AI agents and human users to focus on smaller, well-defined problems.

Alignment with Real Engineering Workflows

The approach mirrors how human designers operate:

- Iterative refinement
- Modular design
- Validation through testing
- Refinement of models remains possible in existing CAD software

This alignment increases the likelihood of adoption in practical settings, as it integrates naturally with existing workflows rather than attempting to replace them entirely.

Extension Beyond Geometry

As demonstrated in earlier sections, once a validated model exists, AI can assist with:

- Bill of materials generation
- Manufacturing planning
- Documentation and rendering

This broadens the value of AI from a design tool to more of a product lifecycle assistant.

5.3 Limitations and Challenges

Continued Reliance on Human Intervention

The system does not eliminate the need for human input. As observed in the bed frame case study:

- Complex geometries remain a beyond the ability of current of AI models.
- Assembly alignment required manual adjustment
- Functional validation depended on user judgement

This suggests that current systems are best viewed as assistive rather than autonomous.

Sensitivity to Design Familiarity

The success of the bed frame example was likely influenced by the familiarity of the design domain. LLMs perform better when generating artefacts that resemble patterns present in their training data.

For more novel or unconventional designs:

- Initial outputs may be less coherent
- Iteration cycles may increase
- Human guidance becomes more critical

This introduces a bias toward known design archetypes and limits generalisation.

Lack of True Spatial Understanding

Even within the iterative loop, LLMs do not develop an internal understanding of geometry. They rely on external feedback rather than reasoning about spatial relationships directly. This can lead to:

- Repeated errors across iterations
- Inefficient convergence
- Difficulty handling complex constraints

Performance and Cost Considerations

The iterative nature of the system introduces computational overhead:

- Multiple model calls per component thus high inference costs
- Repeated CAD execution cycles
- Increased latency for complex designs

5.4 Implications for AI and Engineering Practice

By lowering the barrier to entry, AI-assisted CAD systems have the potential to enable a wider range of users to participate in product design. Individuals without formal CAD training could prototype ideas through guided interaction.

The role of the designer may shift from manual modelling to:

- Defining intent
- Human guided iteration
- Evaluating outputs

This represents a move toward supervisory and conceptual design roles, with AI handling lower-level implementation details, much like what is happening in the software engineering field currently.

Rather than replacing existing tools, AI will augment them, creating hybrid workflows that integrate multiple modes of interaction.

Importance of System Design Over Model Design

Perhaps the most significant implication is that progress in this domain depends less on improving LLMs in isolation and more on designing effective systems around them.

Key areas of focus include:

- Feedback mechanisms
- Orchestration architectures
- Integration with domain-specific tools

This suggests that AI engineering in this context is a systems problem, not just a modelling problem.

---

6. Conclusion and Future Work

This report examined the use of large language models in computer-aided design, focusing on the gap between generative AI and the strict requirements of parametric CAD systems. While AI enables natural-language-driven design, current LLMs cannot reliably produce engineering-grade CAD models independently.

The core challenge is the mismatch between probabilistic AI generation and the deterministic constraints of CAD modelling, which requires geometric validity, consistent feature relationships, and functional coherence. As demonstrated through analysis and the ElixiHub case study, AI-generated outputs still require validation, refinement, and human correction. To address this, the

report proposed an iterative AI–CAD workflow where LLMs generate parametric scripts, CAD engines validate them, and human users guide the process through spatial reasoning and design intent. In this model, AI acts as a collaborator rather than an autonomous designer.

The bed frame case study highlighted both the strengths and limitations of this approach. Although the system generated a coherent multi-component model, human intervention was still required to correct joinery and assembly alignment, and performance may decline in less familiar design domains. Despite these limitations, combining AI generation, deterministic validation, and human guidance enables faster convergence on viable designs. Once validated, these models can also be refined further by experienced CAD designers.

Overall, the findings suggest that the future of AI in CAD lies not in replacing designers, but in augmenting them through iterative, tool-integrated workflows that combine generative flexibility with deterministic correctness and human judgement.

6.1 Future Work

While the proposed approach represents a meaningful step toward practical AI-assisted CAD, several areas remain open for further research and development.

Improved Geometric Reasoning
Current LLMs lack true spatial understanding. Integrating geometry-aware models and specialised LLM “skills” could improve constraint handling, physical feasibility, and first-pass accuracy.

Domain-Specific Training
Fine-tuning on CAD-focused datasets, especially parametric models and assemblies, could improve reliability and reduce human correction.

Lifecycle Integration
The greatest opportunity may lie in extending AI-assisted CAD into full product development workflows, including:

- Manufacturing and process planning
- Cost and supply-chain analysis
- Documentation and compliance

---

7. References

[1] R. Wang et al., “Text-to-CAD Generation Through Infusing Visual Feedback in Large Language Models,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

[2] K. Alrashedy et al., “Generating CAD Code with Vision-Language Models for 3D Shapes,” OpenReview, 2025.

A demonstration of the ElixiCAD application can be provided on request.

contact@realised.io