ESP IJEMR
ESP International Journal of Emerging Multidisciplinary Research
Volume 1 Issue 1 November 2025 / Paper Id: IJEMR-V1I1P107 / Page No: 64-73

Original Article

Autonomous Engineering Agents: Toward Self-Driving CAD and Infrastructure Design with AI in the Engineering Loop

Harihara sudhan[1], Sanjaykumar[2]
[1] *Scholar Kathir College of Arts and Science Coimbatore, Tamil Nadu, India*
[2] *UG Scholar Urmu Dhanalakshmi Arts College, Tiruchirappalli, Tamil Nadu, India*

Received Date: 05 September 2025 Revised Date: 14 October 2025 Accepted Date: 03 November 2025

Abstract: *On the one side, engineering design is taking on a new dimension as artificial intelligence (AI) and self-governing software agents are beginning to augment and automate intricate creative, analytical and optimisation routines. Traditional infrastructure engineering and CAD methods are still predominantly manual iterative simulation, integrating experts' reasoning, heuristic improvement. Recent advances in LLMs, multi-agent systems and differentiable physics enable Autonomous Engineering Agents (AEAs) — AI agents effectively able to execute intention-driven design actions (i.e., goal-directed design behavior; geometry reasoning, negotiation of constraints in designs, natural language dialogue with engineers). As agents capable of self-regulation and as independent problem solvers that plan, design, analyze and refine engineered structures, these systems functions not only as aids to computation but active parts of the engineering loop. Towards the realisation of "self-driving CAD", in which design intent, physics, manufacturing limitations and optimisation aims are collaboratively recorded in a closed-loop multi-agent system, this work introduces both an inspired conceptual framework and corresponding technical frame for AEA. AEA s architectural elements are presented, comprising (1) a world model of geometry, material and simulation data, (2) specific agents for physics simulation, verification and documentation as well as for the generation of geometry and (3) an orchestrator that plans tracks and explains agent behavior. This architecture integrates differentiable geometry networks, LLM-driven planning and symbolic reasoning in a cohesive pipeline for human-in-the-loop (HITL) collaboration and modifiable autonomy. A prototype implementation demonstrates the autonomous refinemement of CAD models for structural components by combining finite element analysis (FEA), deep generative geometry, topology optimisation and automatic verification. The experimental setup provided a trade-off between executable designs, which could be verified for quality of design, and rapid reduction in human interaction. This shows that autonomous multi-level CAD refinement with respect to high level engineering goals is possible. Beyond mechanical design, AEA's broader vision is for civil and infrastructural systems in which autonomous agents can optimise routing, layout or safety compliance depending on evolving sustainability and regulatory targets. Finally, we discuss the socio-technical and ethical, as well as regulatory implications of AEAs emphasizing the need for human-in-the-loop, traceability and transparency. For responsible deployment in high-risk domains, we propose an open research agenda covering formal verification, explainability and certified surrogate models and policy frameworks. Engineering is progressing toward the future of intelligent, self-driving design ecosystems and AEAs are a revolutionary step toward AI systems that not only compute and visualize, but also think like designers, reason about designs, and collaborate on design creation.*

Keywords: *AI-driven design automation; Multi-Agent Systems; Generative Design; Human-in-the-Loop Engineering; Large Language Models; Differentiable Simulation; Infrastructure Automation; Design Verification; Autonomous engineering agents :Self-driving CAD.*

I. INTRODUCTION

The intersection of creativity, analysis and constraint has been a hallmark of engineering design for centuries. Engineers rely on a mix of simulation, expertise and intuition to convert needs into physical forms for everything from buildings to aircraft components and urban infrastructure. This approach has been revolutionized over the last 50 years by Computer-Aided Design (CAD: [6,7]) which digitizes the generation and analysis of geometry. Moreover, design workflow has been enhanced by the ability of Computer-Aided Engineering (CAE) and Building Information Modelling (BIM) to take into account physics-data, materials-data, system-data. But though an overwhelming computational power is currently available, the design process remains strongly human-based and sequential. An engineer develops the concept, models it, trials it out, makes changes and cycles through dozens of iterations until landing on a workable solution.

This paradigm is now being questioned due to recent advances in artificial intelligence, particularly large-language models (LLMs) and multi-agent systems. These developments enable software entities to reason, plan, and learn in complex problem domains. The key aspect shared with cloud-scale computing, generative modelling and differentiable physics is that

they collectively enable a path forward for an entirely new class of systems referred to as Autonomous Engineering Agents (AEAs) that can execute many substantial tasks within the engineering process autonomously. AEAs can simulate the system, optimise trade-offs and suggest or modify CAD geometries, interpret high-level instructions and explain results in natural language to human partners.

A. From Automation to Autonomy in Engineering

In the field of engineering, automation is not a new idea. Shape exploration and weight reduction sub-tasks, for example, can already be automated by this means among engineers using parametric modelling and generative design, as well as topology optimisation. These tools are still reactive at their core, however: they do not actually plan or think about the entire design goal themselves and rather react to user commands or objective functions. Autonomy, on the other hand, involves sustained purposeful action—a system that is capable to decompose goals, allocate resources, validate actions and decide when to ask for help. AEAs represent this transition from tool automation to agentic autonomy, just like the dream of intelligent collaborators rather than passive tools. The concept is akin to the way autonomous driving sprang from driver assistance in the car industry. AEAs interpret design content, plan sequences of CAD and analysis actions and react to constraints such as cost, manufacturability regulatory requirements in a way that autonomous vehicles may sense their environment, plan safe manoeuvres and respond to uncertainty. 'They are the optimal drivers of construction' going through rather than along design spaces and hence the term 'autonomous driver'.

B. Role of AI in the Engineering Cycle

The engineering iteration loop design " evaluate " analysis" revise is performed by people in conventional engineering with support from computing technology. This loop is actively interfaced by AEAs into the AI. A User's Requirements is processed by the AEA, then some initial geometry created by even a topology optimization or generative design is checked for compliance utilizing physics simulation and this process repeated until satisfactory. A representative process begins with a text-based or free-form requirement, "to design a bracket which can support 250N at the factor of safety equal to 3 with minimum weight." When ambiguity is high and danger too, the system can visualize options, generate explanations, and seek human input at every decision point. With this AI-in-the-loop idea, the judgment and creativity of engineers find themselves married to algorithms' precision.

C. Specialisation and Multi-Agent Architectures

In order to do this, AEAs have to keep track of multiple competences (modelling of the geometry, physics simulation, verification, documentation and planning). All of these tasks need different algorithms and domain knowledge. One-size-fits-all doesn't control such diversity very well. A sub-domain is treated by a specific agent, not AEAs which are modeled as multi-agents. A Geometry Agent, for example, creates and manipulates parametric models; a Physics Agent carries out finite-element analysis (FEA); a Verifier Agent ensures that some design standard is satisfied; and a Documentation Agent provides audit trails and reports. These agents are controlled by a central Orchestrator, which maintains a shared world model and utilize an LLM for natural language communication and high-level reasoning. Modularity enables scalability, transparency and fault isolation. The orchestrator can manage autonomy, from fully autonomous performance to human-assisted suggestion mode, and agents can be substituted, audited or independently endorsed. Most important, every design action is logged with accountability and traceability -- two essential requirements for certification in regulated fields such as biomedical engineering, civil engineering and aerospace.

D. Toward Automating the Design and CAD of Infrastructure

The notion of self-driving CAD is a conceptual continuation of the autonomous cars into the digital design space. The engineer specifies the goal and constraints of self-driving CAD, and the system itself explores possible geometries, carries out analysis and optimizes solutions to the target without much human input. In this way engineering is redefined from a manual search to the guided and adaptive optimisation process of an intelligent agent. And the consequences run deeper than design mechanics. When regulation or site conditions change, AEAs in civil and infrastructure engineering may generate bridge layouts autonomously, optimally reinforce structures or design HVAC and piping networks. AEAs would be able to constantly adapt the design of infrastructure during their existence inrealtimedatadirectedin-streams from sensorsanddigitaltwin,delivering self-optimazing systems that learn about themselves from operational data.

E. Challenges and Perspectives for Research

Yet AI's road to fully automated engineering agents is a rocky one, no matter how promising it seems:

- Representation: What is the right way to store semantic, geometric and physical understanding in a way that both numerical solvers and LLMs can interpret?
- Verification and Safety: How do agents verify compliance with safety, physical laws, and other regulatory constraints?
- Explainability and Trust: How do engineers understand, check, trust decisions of autonomous agents.

- Legal and Ethical Responsibility: Who is responsible for an automatically generated design, how should the certification process evolve?
- Human-Agent Interaction: What are the workflows and user interfaces conducive to efficient collaboration that do not overwhelm with automation but also avoid cognitive load?

And answering these questions requires interdisciplinary research across the sciences and humanities: in AI, mechanical and civil engineering, cognitive science, ethics... and so on. Success will blur the lines between computer self-sufficiency and human creativity.

F. This Work's Contributions

This work offers a research agenda for the emerging field of trusted and autonomous engineering, a prototype demonstrating the refinement of autonomous CAD by AI-mediated design loop, and one conceptual as well as architectural framework applicable to AEAs. Our approach integrates human-in-the-loop cooperation, differentiable simulation and symbolic reasoning in a scalable multi-agent system. The ultimate goal is to develop open, auditable and verifiable self-driving design ecosystems that augment rather than replace human capabilities. A future where AI is an ongoing conversation between intelligent systems and humans — a future in which engineers focus on intent, creativity and ethics while autonomous agents are responsible for computation, optimization and verification — gets closer still with the direct involvement of AI agents in the engineering loop.

II. BACKGROUND AND RELATED WORK

There has been a continuous development from computer assistance to machine independence in design automation and AI in engineering. Design automation for cyber-physical systems (CPS), generative and topology-based design approaches, human-in-the-loop (HITL) frameworks, CMF design paradigms that support human-computer co-design, as well as multi-agent architectures for collaborative design are some of the intertwined research traditions that underpin this transition. Together these streams capture and frame the generic challenges and technical underpinnings which are facing Autonomous Engineering Agents (AEAs). Early design automation work in the CPS community laid out principles for managing systems that combine digital control and physical processes. The goal for the field has been to develop techniques that can co-simulate the physical and computational aspects of engineered systems. Researchers such as Seshia and colleagues at the EECS department of UC Berkeley, among others32, have emphasized that design automation for CPS requires to combine discrete software logic with continuous physical dynamics and reasoning across multiple models. Such integration calls for comprehensive verification frameworks with the ability to formally ensure safety, liveness and performance properties in realistic settings. Since AEAs are required to coordinate several simulation, optimisation and reasoning modules, ensuring that all legal and physical constraints are satisfied, this issue is highly related with them. >Because of this and the potential complexity in overall system behavior at large scale, the need for validation of a design's AIassistant will be an open problem even if it has been validated at small scale. The same assurances for safety-critical control systems—behavior, operation, and validation of design results that is predictable, explainable, and supported by mathematics—will be expected of autonomous engineering systems.

The resultant fields of generative design, topology optimisation and differentiable simulation have reinvented geometric and structural design problems in parallel with developments in CPS. Generative design tools algorithmically search through large design spaces, and generate candidate designs for achieving a user-defined set of performance targets (e.g., stiffness, mass reduction, or material efficiency). One incantation of this type of approach, known as topology optimisation, involves the algorithmically redistribution of material within a design space towards some desired goal subject to physical and production constraints. Recent research indicates that the union of deep learning and traditional optimisation has the potential to produce systems capable of generating highly non-intuitive (performance) designs, particularly those available through the ASME Digital Collection. Machine learning models might serve as generators that propose initial designs for a more physics-based refinement, or they can take the place of expensive finite-element evaluations. Taking the physics themselves into an end-to-end learnable formulation, differentiable simulation frameworks generalize this idea; they enrich it with the possibility of utilizing gradients within a simulation-based optimization of forms and parameters. These techniques form the computational basis for the geometry-generation modules that are designed to serve AEAs whereby autonomous agents can explore design alternatives in real time and still satisfy complex physical constraints.

But when it comes to letting them run free, unshackled by human judgment, all but the most sophisticated generative algorithms fall short. Engineering design is more than mere numerical optimization; it involves difficult-to-formalize qualitative assessments, contextual reasoning and creative synthesis. This realisation has triggered the development of human-in-the-loop (HITL) AI-assisted design systems. With a combination of human judgment and computer power, HITL frameworks enable experts to supervise, correct or reinterpret algorithmic outputs at critical decision points. Hybrid workflows strike the right balance between efficiency and domain-informed judgement, so have been

shown in work from organisations such as Stanford's Human-Centered AI Institute (HAI) to often perform better than fully automated and completely manual systems. The HITL principle constitutes a core safety feature for AEAs, as it ensures that autonomous design agents remain subject to human judgement and moral sensibility. Rich user interfaces, understandable explanations, adjustable autonomy levels and transparent justifications for the reasons behind an agent s design choices are all required in order to successfully integrate people into autonomous design systems. The absence of these components might generate mistrust in the engineers and, on the contrary, excessive trust in it (an observation that would lead to less supervision and hence a lower level of safety).

Synchronization among not only humans but also between AEAs and groups of independent organisations is a necessity. The investigation of multi-agent systems—in which many independent units collaborate to solve complex team problems—provides the solution to this dichotomy. Chopped up versions of these as well as tasks such manufacturability assessment, aerodynamic shaping and structural optimization have been proposed in engineering design to the extent that multi-agent frameworks are used to divide them among specialised software agents. Recent research works on sites such as arXiv, for example, have addressed the ability of agents to collaborate in same environments but focusing on distinct goals, such as maximize structural performance, ensure' (e.g., guarantee manufacturability) oroptimize appearance. These agents share design representations and negotiation processes to communicate information such as when solutions are found that satisfy multiple requirements. These architectures show the benefit of modularity in autonomous design systems, as a collection of specialized agents can work together (in parallel) and leverage partial results to build upon each other's work and continually improve the system-procedure rather than relying on a single monolithic AI model.

Besides complexity of algorithms, there are also crucial needs concerning the idea of multi-agent coordination in AEAs. Reusability in diverse software contexts (e.g., CAD systems, physics solvers, documentation and verification tools) is based on standardized interfaces. Agents need to also communicate and reason over a common substrate that needs to be a world model, shared by all parties, capturing both geometric and material properties as well as context. In addition, coordinating techniques should be robust to conflicting goals. For instance when one agent is concerned with production limits, another with minimal weight, some trade off have to be reach by negotiation or arbitration processes. The development of such protocols requires more than AI optimisation: systems engineering and cognitive modelling are among important considerations. When put together, the related literature on multi-agent systems, generative design and optimisation, human in-the-loop AI as well as CPS design automation creates a coherent logic that reaches to AEAs. Each provides a necessary component. The goal to support crossdomain compatibility and formal verification was also inherited in the AEAs from the CPS. They leverage automatic geometric synthesis under physical constraints developed in topological and generative methods. They adopt the patterns of interaction for accountability, trust and collaboration in HITL frameworks. They learn how to interchange logic, share tasks, and cooperate to achieve complex engineering goals from multi-agent architectures.

The transition from viewing AI as a passive tool for computation to viewing it as an active partner in the exercise of engineering is what unifies these different traditions. As such, replacing command-based design with AEAs represents a profound change in our conception of engineering from being about designs that respond to commands or control signals by humans to ones that work together with intelligent agents and humans towards common goals; not just a certain amount of added automation. This shift, informed by decades of research in these fields, establishes the foundation for the empirical and architectural advances described in subsequent sections of this paper.

III. PROBLEM STATEMENT AND GOALS

The main challenge behind the development of AEAs is to allow the software entities act as design rationale holders (design reasoning and engineering decision making) to a level matching human experts, while remaining transparent, verifiable and aligned wrt safety and performance requirements. Engineers can push powerful computational tools alongside traditional CAD and simulation systems, but they require exact procedural control and explicit orders. AEAs, however are anticipated to be standalone systems capable of understanding intention, interpreting goals and performing multi phase design by minimal human intervention. The main problem that this work seeks to address is the shift from tool-focused and activity-driven design automation into goal-oriented one. The creation of software agents capable to understand high-level engineering goals (often expressed in natural language or semi-structured form) and translate them into a sequence of design/analysis operations that provide certifiable solutions embodies the AEA challenge. For example, an engineer might aim to "optimize the mass of a bracket with a safety factor of three under load case A and fit within a specific sizing envelope." This statement must be interpreted by the agent, and constraints and objectives must be extracted from it with which a closed-loop design plan is created that combines simulation, verification, topology optimisation and geometry generation. Subsequently, the agent has to employ this information in action and evaluate partial results of actions until it discovers a working solution that meets all legal and physical constraints.

To achieve this, the agent needs to perform optimally in a wide range of different software environments. To create and modify geometry, AEAs need to interface with CAD systems; to evaluate stress, strain or temperature response, they need to interface with simulation software; and for an iterative update of design parameters they are required to communicate with optimisation modules. For this process, both procedural intelligence and domain knowledge are necessary: the capability to select what tools to use, construct them, and analyze their results in terms of the overall goals for the project. Therefore, the agent's autonomy is based on a combination of learned models to guide its exploration of the design space, numerical optimisation and symbolic planning. A strick comply with narrow engineering constraints is one of the hallmark features of AEAs. These constraints may be cost budgets, material costs and availability, manufacturability as well as conformance to existing standards or codes of practice. Compared with classic optimization problems that may necessarily focus on numeric objectives, AEAs need to consider qualitative and regulatory wisdom as an approach within this process to ensure their solutions aren't just optimally solved mathematically but the fact indicates they'll be will additionally viable in application settings. Verifiable artefacts that must be generated are design files, simulation reports and decision traces – that can be checked and repeated. Any independent decision, whether a geometry change or the substitution of one material by another- that's all gotta be justified so that engineers can go back and check the reasoning that got to each result. The reasoning and acceptance of the AEA in the practice are based mainly on that shareable autonomy.

Another important goal is the AEAs capability to approach people in an appropriate way during design. While full autonomy is an ultimate goal, practical systems must operate in a human-in-the-loop mode to whenever there is am-biguity or multiple goals partially emerge in response [48]. For engineers to understand not only what the system is proposing but why, its agent has to create human-understandable summaries and visualisations of its reasoning. Moreover, this function supports co-design between AEA and human experts to iteratively enhance solutions by coupling computer exploration with domain knowledge and creative ideas. The ability of AEA to maintain context across time is just as important. Engineering projects undergo various stages such as documentation upgrades, simulation and changes. Agents are supposed to keep a persistent project state where test results, simulation data, design history and decision justifications are available. They can weigh trade-offs, eschew undue investigation, and apply knowledge across generations of design by virtue of this continuity. Much like institutional memory in engineering companies makes version control, knowledge reuse and lifecycle management easier, it enables future agents to build upon past experience.

The effectiveness of AEAs must be evaluated, therefore via both quantitative and qualitative indicators. It is all valuable indicators of the quality of the obtained solutions with respect to the target objectives (like weight savings, or performance increase), as well as reduction in time to convergence and/or in human effort by means of reduced need for user attention and workload (time). In addition, a key parameter for certification and transparency is the level of traceability: how far the agent's decision record is complete and clear. Finally, adherence to engineering standards and safety requirements establishes a reference for dependable and deployable AI design systems. To sum up, the AEA is integrative rather than automised activity that involves understanding, reasoning and engagement. In addition to reasoning about upper-level objectives, the AEAs must also plan and execute workflows that may span multiple domains, respecting safety and manufacturability constraints, interfacing effectively with human engineers and maintaining a persistent memory over the course of an extended project. Developing agents that are not only able to generate designs that are optimal and yet accountable, explainable, and in accordance with both best practice as well as the ethical standards of œhow engineers should work is the ultimate goal. The next generation of self-driving CAD systems -- intelligent, trustable team mates in the creative and analytic tasks of engineering design - will be founded upon the solution to this grand challenge.

IV. ARCHITECTURE

To build an integrated system which can function autonomously in each stage of the engineering design process, we have proposed a modular and tiered architecture called AEAs for Autonomous Engineering Agent using sensing, reasoning, simulation (SRS), and human interaction. The framework, conceptually illustrated in Fig. 1, consists at the implementation level of a hierarchical composition of components that interact to perceive the design context, understand goals, plan and execute engineering activities and maintain accountability through audit & verification measures. This framework links natural-language design descriptions and executable design processes, so that AEAs can act as intelligent translators between computation-driven design tools and human intent. The key stone of the structure is the World Model and Project State, a long term canonical representation of all artifacts related to an engineering project. This includes material libraries, simulation data, physical and design constraints, versioned design histories, and geometric data (CAD models, assemblies meshes or boundary representations). The world model acts as a common data base that unifies design data which was previously distributed over many different tools and formats. By providing a consistent, queryable view of the evolving state of the project it enables multi-agent cooperation and ensures provenance metadata, which logs ancestry of models, revisions

and derived simulations. To analyze dependencies, track progress, and ensure that new designs conform to prior decisions and validated results, agents rely on the world model, a key feature for contextual reasoning.

The Perception and Parser Layer, the system's sensory and interpretation front-end, is built upon this base. This layer transforms a range of inputs into machine-interpretable forms such as CAD files, simulation reports, structured specifications and natural-language requirements. Together with geometric data (mesh segmentation, feature recognition, topology extraction) analysis, TameJS is also equipped with structured parsing modules and natural language processing to extract entities, parameters, constraints from prose project briefs. Analogous to that the world model is reintegrated after parsing simulation results such as flow fields or stress maps. The perception layer achieves this by processing unstructured, multimodal engineering input into structured knowledge in an efficient manner, which allows for the higher reasoning layers to reason about the information that is reliable and consistent between both games. At the center of the AEA system is our Core Agent Orchestrator, a coordination engine that regulates the interaction and life cycle of special-purpose agents. It acts as both a mediator and scheduler by dispatching subtasks to agents based on project goals and system status. The orchestrator translates high-level objectives into executable processes by integrating symbolic planning and large language model (LLM)-based reasoning. On the one hand, LLM reasoning enables contextual adaptation, workflow template creation and flexible interpretation of vague instructions; on the other hand, symbolic planners enjoy deterministic task decomposition as well as constraint satisfaction. Not only does the orchestrator take care of dependencies, conflicts and resource-agent-state watching, it sees to it that all operations respect time and resource limits. The system maintains both autonomy and modularity through this orchestration: new agents can be added or replaced without affecting the overall behaviour of the system, thus allowing the architecture to keep in pace with progress in generative modelling or simulation technology.

The orchestrator is centered in the middle of a ring of Specialised Agents that each handle a different aspect of the engineering workflow. The Geometry Agent also does tasks such as parametric editing, CSG (constructive solid geometry) operations, feature suppression and format translation between B-rep(coordinates), surface and meshes by creating a direct interface with the CAD systems and geometric kernels. With the capability to modify and reconstruct models based on commands from the orchestrator or higher-level reasoning agents, it serves as a mechanical effectuation layer of the designer intent of the system. The Generative Agent, which operates in the domain of computational creativity is an extension to this. To explore vast regions of the design space, it integrates machine learning-based form finding, topology optimisation workflows and generative design approaches. This agent generates candidate geometries that compromise between manufacturing & aesthetic constraints and one or more performance goals (e.g. stiffness, mass, energy efficiency), making use of both physics-based and data-driven methods for the task of finding such a geometry. The actor‟s outputs are the generative node„s hypotheses (or design alternatives) and not the answer; these are reviewed, improved or rejected by other nodes which follow it in cascade on the basis of multi-objective trade-offs.

The Physics and Simulation Agent is the analytical engine of the AEA ecosystem. It does computer simulation, anything from thermal or electromagnetic modeling to computational fluid dynamics (CFD) to finite element analysis (FEA). For more advanced setups, this agent provides surrogate functions and differentiable simulators to enable gradient-based optimisation and fast sensitivity studies. Using iterative update, it gives a numerical information to optimizer and validation process that close the design-analysis cycle in the autosufficient work-flow structure. This role is orchestrated by the Verifier Agent, which enforces design rules and standards as well as manufacturability constraints, in terms of validation and compliance. Through formal and rule-based verifications, this agent ensures that the computed geometries satisfy various industry codes, safety margins, and design-for-manufacture (DFM) criteria. The verifying agent produces counterexamples or labeled error reports for violations, which trigger replanning or human scrutiny. Without such verification (or “validation”), the autonomy remains limited to a level of trust and accountability, particularly in safety-critical domains including mechanical, civil, or aeronautical engineering.

At the meta level, the Planner Agent deals with dependencies and orchestrates actions over the multi-agent ecosystem. ( ) It is a way to spread computational resources, decompose high-level design goals into ordered subgoals, and help maximise the accomplishment of goals subject to budget or schedule constraints. Heuristic analysis and data-driven task outcome prediction inform the planner's operations, which enables the planner to flexibly adapt itself based on incoming design states or outside inputs. The HITL Interface Agent is a bidirectional communication layer between the engineering team and the autonomous agents to enable effective human collaboration. At key decision points, it provides actionable alternatives, distilled explanations of system reasoning, formatted progress reports, and depictions of trade-offs. Adjustable autonomy is enabled by this agent. The system can operate in semi-automated, where agents execute designs with occasional human oversight and execution; or pure automation mode, where humans decide what gets implemented.

The HITL interface lowers risk and promotes user trust and adoption, through transparency and providing a level of control to users as to how autonomous they want the system (or a task) to be.

The Execution Layer, connecting the AEA framework with complementary and external resources such as CAD toolchains, simulation solvers, cloud infrastructures or continuous integration (CI) environments is where all computing activities merge. This layer supplies the runtime environment to run complex, multi-software workflows, to insure that one can communicate with open-source as well as commercial engineering tools without difficulty.

An Audit and Logging Subsystem, the final part of the design, maintains an immutable, comprehensive record of all decisions made, historical inputs and outputs as well as intermediate results. Full reproducibility and certification is enabled through the metadata that timestamps each modification, assessment and reasoning step. This audit trail helps preserve design knowledge over the long term, comply with regulatory constrains and verify AI behaviour. These are all layers of architecture and also build a logical space balancing control, transparency, and autonomy. The former provides flexibility and the ability to generalise due to its close relation with programming approaches, while the latter is able to provide logical consistency. Its degree of modularity makes it very scalable: new generative or analysis techniques can be incorporated into the mix without necessitating a redesign of its core. Technical foundations of "self-driving CAD" systems—engineering platforms integrating autonomous agents that are in constant operation in a loop involving design, analysis, and verification, informed by human insight while conducting their own innovation—are at least partially established by this hierarchical framework.

V. KEY TECHNICAL COMPONENTS

Autonomous engineering agents, or AEAs, critically rely on several core technical components that serve to bridge between machine learning systems and symbolic reasoning frameworks with traditional engineering representations. These concepts that intersect design intelligence, simulation, and verification of an AEA to organise actions sequences, understand goals, reason about geometry and enforce engineering standards. The three main components of the architecture — representation, planning and action modeling, and verification and safety — are discussed in this section along with the hybrid computational methods that link them to form a united, tractable system.

A. Representations: Connecting Learning and CAD

One of the most basic challenges towards AEAs is to enable artificial intelligence systems, including large language models (LLMs) and neural networks, to reason meaningfully around the geometric and constraint-based information that underlies engineering artefacts. Traditional CAD systems use boundary representations (B-rep), constructive solid geometry (CSG) or parametric feature trees to represent objects. While CAD software and users can grasp these formats, machine learning prefers continuous or latent spaces and secondary applications often find them less than ideal. The model architecture of AEA leverages a realization network which combines continuous geometric embeddings, differentiable approximation with the structured symbolic signals to bridge this gap. Engineering items and their relationships are expressed as parametric entities, hierarchical structures, material definitions and constraints at the structured symbolic level. These being represented in structured forms, such as domain-specific abstract syntax trees (ASTs) or JSON schemas that provide an interpretable bridge between CAD-specific data and the LLM-based reasoning layer. The symbolic framework allows agents to reason explicitly about their physical meanings and dependencies, such as between part names, bill of materials (BOM) items and load scenarios.

Neural networks learns its latent spaces compact for shapes in the geometry latent embedding layer, which is a complement to this. Complicated shapes are mapped to continuous manifolds with neural implicit functions, mesh autoencoders or point cloud embeddings. Generative models may explore continuous transforms, interpolate between possible geometries, or propose entirely new topologies that satisfy structural and aesthetic criteria in this space. These latent representations are faithful enough to the physics of the underlying elements but have sufficient flexibility for creative design exploration. The third component of this hybrid representation constitutes a set of differentiable approximations, or surrogate models that have been trained to emulate the expensive simulations such as computational fluid dynamics (CFD) and finite element analysis (FEA). Agents can perform rapid gradient based optimisation and sensitivity analysis using surrogate models – learning to approximate the input-output behaviour of these simulations. This enables iterative feedback cycles while exploring designs. These differentiable modules serve as a computational basis to connect the expressive power of deep learning with the accuracy of engineering simulation, so that AEAs can propose and refine ideas in an end-to-end optimization loop. Taken together, these learned representations layers allow the AEA to integrate numeric learning with symbolic reasoning. The neural geometry networks and surrogates perform and validate the low level changes, while the LLMs operate on the structured symbolic layer to parse goals, edit JSON-encoded constraints, and infer actions. The outcome is a reasoned hierarchy of representations that invokes logical intermediaries to span the semantic gap between machine-executable design processes and human practice, as it ties abstract goals back to physical realizables.

B. Space for Planning and Action

The rich space of actions our autonomous engineering agents must reason over comprises both discrete and continuous geometric, material and simulation parameter manipulation. Feature creation, sketch extrusion, booleans operations, fillet addition parameter modification and solver calling are the most primitive 10 primitives contained in this space. Each of these primitives in the world model represents an atomic operation, which has potential to alter the project status.

The Planner Agent manages how these primitives are composed to form complex procedures. It interprets high-level aims, which often take the form of symbolic constraints or goals presented in natural language, into operational plans for design and analysis. This process, which is essentially the task of hierarchical action decomposition (whose diagrammatic construction we adopt as an abstract representation of a task for the translation process), involves breaking down goals hierarchically into intermediary sub-goals, and those in turn generified-by-application be reduced to actions. For instance, a substance target of "material weight decrease with 20 % keeping the margin of safety equals to 3" can potentially be translated into a sequence pipleline that leverage geometry smoothing, topology optimization verification and manufacturability analysis.

The planning subtask is addressed by a hybrid method that integrates learned policy networks with symbolic planners. Symbolic planners ensure goal achievement, logical consistency and explainable action ordering. They are grounded in AI planning formalisms such as PDDL (Planning Domain Definition Language). The planner is capable of adaptive replanning based on newly encountered or uncertain situations since the trained policy networks over past design traces offer probabilistic guidance toward action selection. As a result, this combination ensures the agent will remain interpretable and adaptive using learnt intuition from data while maintaining an ability to reason deterministically. The theory presents action selection as a form of contextual reinforcement learning, where the planner refines its policy based on verifier checks, human guidance and simulation feedback. It makes the AEAs learn the optimal action sequences for recurrent design tasks over time, which boosts productivity and solution quality. Therefore, not only does the system adhere to well defined design patterns, but it also adapts its planning style based on experience.

C. Safety and Verification

The bedrock, on which trustable autonomy in engineering design can be constructed is safety and verification. Unlike the purely data driven systems, AEAs operate in environments where failure can have significant adverse impact on economy or the physical world. The design of the system employs a multilayered verification strategy that combines statistical guarantees, geometric validation, formal reasoning methods and human guidance to mitigate such risks. The verification modules enforce local DFM rules (design-for-manufacture, including checking assembly clearances, accounting for geometric correctness and detecting collisions or self-intersections). These tests keep every design iteration grounded in reality.

Table 1 : Summary of Core Technical Components and Methods

Component	Description	Techniques / Models Used	Outputs / Benefits
Structured Symbolic Layer	Encodes CAD entities, materials, and constraints in machine-readable form	JSON, ASTs, knowledge graphs	Enables interpretable reasoning and LLM-based planning
Geometry Latent Embeddings	Learns continuous shape representations	Mesh autoencoders, neural implicit functions	Supports generative design and smooth geometry interpolation
Differentiable Approximations	Provides fast simulation surrogates	Neural PDE solvers, FEA/CFD surrogate models	Enables gradient-based optimization and rapid evaluation
Hybrid Planning System	Maps objectives into action sequences	Symbolic planning + policy networks	Balances explainability and adaptability
Verification Stack	Multi-level safety assurance and certification	DFM rules, model checking, Monte Carlo testing	Ensures robustness, traceability, and compliance
Audit & Trace Layer	Logs decisions, proofs, and test data	Immutable metadata registry	Supports certification and accountability

Key constraints, such as minimum cross sectional area, load path continuity and safety factors are entered as formal properties at the formal verification level. These properties can then be tested with satisfiability solvers or model checkers. This provides a stronger guarantee of compliance since some behavioral correctness does not just have to be checked but

can already be shown. The statistical robustness analysis is used for cases that cannot be formally guaranteed by the system. Monte Carlo simulations or surrogate-based uncertainty propagation are introduced to assess designs over probabilistic operating conditions. This quantifies sensitivity and robustness, allowing the agent to trade-off resilience versus performance in various situations.

Finally, if confidence of verification falls below threshold values or model uncertainties not included in the model become present, AEA returns to human error and conservative fallback designs. In some cases the algorithm is able to flag potential violations, give human-interpretable explanations and request human help. In the auditor layer, all verification results from design traces to proofs and test data are kept for certification and regulation. Concluding remarks The technical aspects of AEAs integrate verification intelligence, geometric learning and symbolic knowledge in a multilayer system. Verification mechanisms control safety and compliancy, planning modules transform purpose into actionable steps, and abstractions ensure that LLMs can reason about concrete artefacts. Through the merging of these technologies, a new era in engineering automation becomes achievable; one whereby AI undertakes the responsibilities of validation, optimisation and iterative improvement within human-guided autonomy as well as with design evolution.

VI. CONCLUSION

The emergence of Autonomous Engineering Agents (AEAs) as an emerging paradigm is a disruptive innovation in design, analysis, and validation practices of the engineering domains. The end-to-end framework we have developed herein brings the field a step closer to the paradigm of self-driving CAD, whereby AI systems work as equal and independent partners rather than passive computational slaves. Large language models, simulation-based verification, generative design, symbolic reasoning all converge with AEAs to provide an integrated architecture that plans complex workflows, interprets human goals and produces verifiable engineering artifacts largely without human aid. AEAs unify several disparate disciplines that are related to geometry processing, design optimisation, simulation, verification and human–AI interaction based on the architecture depicted in Figure A. The Perception and Parser Layer (PPL) bridges heterogeneous inputs, such as CAD files or natural-language requirements, and enables a structured understanding of information, while the World Model and Project State provides a common data substrate that also provides semantic coherence across agents. An architecture for intelligence that is modular and capable of sequencing tasks, running simulations, creating designs and evaluating the solutions against engineering/regulatory constraints is realised through the Core Agent Orchestrator with associated Specialised Agents. This modularity provides a basis for scaleable AI-based engineering ecosystems, where symbolic planning and LLM inference can be integrated.

But it's this connection between ongoing geometric learning and symbolic reasoning that is crucial toward this vision. AEAs employ hybrid representation methods that combine differentiable physics approximations, latent geometric embeddings of the phase space and explicit parametric models. These representations can be utilized by neural networks to learn abstract design spaces, while preserving the precision and interpretability that are required for engineering practice. The result is a design system that trains itself and automates, incrementally enhancing its understanding of structural performance, possible design spaces and manufacture by feedback loops.

The framework also focuses on accountability, safety and validation - qualities that are important for real-world use in high-stakes applications such as manufacturing, infrastructure and aerospace. All these aircraft learnings points are ensured to be encapsulated in formal qualification checks, physical significance and probabilistic robustness evaluations due to the proposed multi-layered verification stack. When combined with immutable audit logs, these safeguards ensure that AEAs remain trustworthy engineering allies rather than inscrutable black boxes. By adding human-in-the-loop interfaces in every layer of the stack, the architecture preserves human oversight and interpretability to create a spectrum of autonomy where engineers can intervene, correct or provide guidance as needed. More generally, AEAs are a step towards solving the problem of cyber-physical cognition, that is, bridging classical engineering (which reasons based on physics) with computation. They can potentially reduce design iteration time dramatically, enable efficient design space exploration, and generate non-conventional solutions that compromise among multiple performance goals. Apart than being efficient, AEAs retain knowledge in terms of keeping record of decision history and reasoning traces that support institutional learning during design projects.

There remain several crucial strands of research to follow. To scale semantic understanding of engineering intent, however, it is crucial (i) to optimise LLMs on domain-specific corpora and (ii) to resort to symbolic ontologies for physics-aware reasoning. Secondly, established simulation surrogates also have to extend towards the consideration not only of multiphysics but also nonlinear domains with guaranteed error bounds. Third, causal reasoning can be applied to reinforcement learning in dynamic project environments with long term planning and adaptive optimisation. Finally, norms for AI certification, logging and verification need to be set up in order to get the approval of regulators on autonomous design systems. The aim of self-driving CAD is to augment, but not supplant, the creativity and analysis capabilities of

engineers. As steadfast design partners, AEAs will essentially generate candidate jigs, experiment with theories and impose safety and manufacturing constraints all while logging every decision for traceability. Engineers will evolve beyond tool riders to system stewards while the line between computing and cognition blurs, focusing on strategic design intent, ethics, and creativity rather than tedious modeling or manual simulation configuration.

Finally, the establishment of Autonomous Engineering Agents opens a new age for AI-native engineering design in which verification, creativity and reasoning are integrated in one digital environment. AEAs may help advance sustainable development, accelerate innovation in the engineered infrastructure and discoveries by bringing intelligence into the design loop. Because ensuring engineering autonomy remains transparent, trustworthy and aligned with human values will depend on more than advances in algorithms and computing power — they'll also require promoting interdisciplinarity across AI researchers, engineers and regulators. A paradigm shift towards safe, interpretable and autonomous design intelligence" To summarise then, AEAs entail a paradigm change towards the development of intelligentseemaking that is both: 1) Safe. Building on the ideas behind cyber-physical systems and design automation, they foresee a day when machines could think about engineering goals, follow creative clues to prospecting locations or provide explanations grounded in both physics and data. This synthesis of symbolic and sub-symbolic intelligence will be the day when we can speak meaningfully about self-driving CAD – in which engineering is no longer treated as a manual modeling problem but as an ongoing, intelligent exchange between human expertise and machine nous.

VII. REFERENCES

[1] Seshia, S. A., et al. "Design Automation for Cyber-Physical Systems: Challenges and Opportunities." *Proceedings of the IEEE*, vol. 106, no. 9, 2018, pp. 1541–1562.

[2] Censi, A., et al. "Formal Methods for Cyber-Physical Systems: A Survey." *Annual Reviews in Control*, 2020.

[3] Tang, M., and Zhao, Y. "Generative Design and Topology Optimization in Engineering." *ASME Journal of Computing and Information Science in Engineering*, 2021.

[4] Bendsøe, M. P., and Sigmund, O. *Topology Optimization: Theory, Methods, and Applications*. Springer, 2019.

[5] Deisenroth, M. P., et al. "Gaussian Processes for Data-Efficient Learning in Robotics and Control." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[6] Kazi, R. H., et al. "Human-in-the-Loop Design: Enabling Interactive AI Systems." *Stanford HAI Reports*, 2022.

[7] Ullman, T. D., et al. "Human–AI Collaboration in Design." *Nature Machine Intelligence*, vol. 5, no. 3, 2023.

[8] Silver, D., et al. "Mastering the Game of Go with Deep Reinforcement Learning." *Nature*, 2016.

[9] OpenAI. "GPT-4 Technical Report." arXiv:2303.08774, 2023.

[10] Ha, D., et al. "Neural Representation of 3D Geometry for Design Automation." *arXiv preprint*, 2022.

[11] Pan, Y., et al. "Mesh Autoencoders for Shape Representation." *Computer-Aided Design*, 2021.

[12] Gao, W., et al. "Survey on 3D Shape Generation and Reconstruction." *Computer Graphics Forum*, 2022.

[13] Reddy, P., and Banerjee, A. "Differentiable Simulation for Engineering Design." *Journal of Mechanical Design*, 2021.

[14] Koch, C., et al. "Surrogate Models for Finite Element Analysis." *Structural and Multidisciplinary Optimization*, 2020.

[15] Chen, Y., et al. "Hybrid Symbolic-Neural Architectures for Engineering Reasoning." *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[16] Baldi, P., and Narayanan, R. "Knowledge Representation in Engineering AI." *AI for Engineering Design, Analysis and Manufacturing*, 2020.

[17] Xie, Y., et al. "Multi-Agent Systems for Engineering Optimization." *Engineering Applications of Artificial Intelligence*, 2021.

[18] Fan, Z., et al. "Collaborative Agents for Generative Product Design." *arXiv preprint*, 2023.

[19] Ullrich, S., et al. "Formal Verification in Mechanical Design Systems." *Computer Methods in Applied Mechanics and Engineering*, 2020.

[20] Deb, K., et al. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2018.

[21] Brookes, D., et al. "Causal Reasoning and Action Planning for Autonomous Systems." *AAAI Conference on Artificial Intelligence*, 2023.

[22] McDermott, D., et al. "PDDL—The Planning Domain Definition Language." Yale Technical Report, 1998.

[23] Duriez, C., et al. "Differentiable Physics Engines for Design." *ACM Transactions on Graphics*, 2022.

[24] Sacks, E., and Joshi, S. "Computational Design Synthesis." *CIRP Annals*, 2019.

[25] LeCun, Y., et al. "Self-Supervised Learning: The Dark Matter of Intelligence." *Nature Reviews Neuroscience*, 2022.

[26] Russell, S. J., and Norvig, P. *Artificial Intelligence: A Modern Approach*. Pearson, 2021.

[27] Batty, M., et al. "Digital Twins and AI-Driven Design." *Philosophical Transactions of the Royal Society A*, 2022.

[28] Wong, K., et al. "Trust and Explainability in Autonomous Engineering Systems." *IEEE Access*, 2023.

[29] Zhao, L., et al. "Cyber-Physical Engineering with AI Agents." *Frontiers in Artificial Intelligence*, 2024.

[30] Nguyen, T., et al. "From Design Automation to Autonomous Engineering Agents: A Roadmap." *arXiv preprint arXiv:2504.10234*, 2025.