ASME Journal of Mechanical Design
Online journal at:
https://asmedigitalcollection.asme.org/mechanicaldesign

Developing Design Theory Using Large Language Model Agents: A Case Study of C-K Theory

Kevin Ma[1]
Department of Mechanical Engineering,
University of California, Berkeley,
Berkeley, CA 94720
e-mail: kevinma1515@berkeley.edu

Eric Reynolds Brubaker
Grado Department of Industrial and
Systems Engineering,
Virginia Tech,
Blacksburg, VA 24061
e-mail: bru@vt.edu

Kosa Goucher-Lambert
Department of Mechanical Engineering,
University of California, Berkeley,
Berkeley, CA 94720
e-mail: kosa@berkeley.edu

Computational simulations have long been used for design and engineering analysis, but there is also a rich history of using simulations to develop design theory. The present work draws inspiration from design, management, psychology, and other fields that have used simulations to develop theory and integrates recent advances in large language models (LLMs). While design theory has been used to improve the development of generative design tools, this study goes the other direction and uses generative AI-based simulations to further develop design theory. We take C-K theory as a case study to demonstrate this approach. The simulations used computational agents fueled by large language models designed to adhere to the C-K theory methodology in both wording and framework. These were evaluated using a mixed-methods approach across three studies, including two simulation experiments and one conventional content analysis of qualitative, LLM-generated, C-K transition rationale statements. The results reveal that concept-to-concept transitions were the predominant C-K operation and that concept diversity trended downward across various experimental conditions. These results are in contrast to the generic generativity described in human-based C-K theory and highlight gaps between C-K as a design theory and C-K in a generative AI-based computational simulation form, suggesting future directions for operational and theoretical development. Ultimately, this case study demonstrates that design theories can be simulated computationally using LLM agents, albeit with differences and potential limits when compared to humans, thus providing the design research community with a new approach to further developing design theories.
[DOI: 10.1115/1.4071235]

Keywords: design theory, theory development, C-K theory, LLM agents, AI, design theory and methodology

1 Introduction

Much work in the engineering design research field involves developing simulation approaches to support design activities. Examples include new finite element methods for structural analyses or new computational fluid dynamics models to support the design of fluid flows, e.g., Refs. [1–5]. However, simulations can be used to support not just design activities or processes but also the development of design theory—the abstraction of results about how design is conducted in a particular situation that provides knowledge of fundamental constructs, relationships, mechanisms, and bounding conditions that can be applied beyond the particular context and study [6–8]. There is a history of using simulations to understand design theory that dates back to at least the 1970s with authors like Newell and Simon who simulated the problem space to understand how different heuristics might navigate it and extended those findings to speculate about human decision-making processes [9]. This is built upon Simon's seminal work, *The Sciences of the Artificial*, whereby he proposed that systematic procedures and logical processes are core components of the design process [10]. However, subsequent works have critiqued these views as unrealistic, noting that design is not entirely rationalistic and frequently influenced by context and situational factors, e.g., Refs. [11,12].

Design researchers have used a variety of simulation approaches to develop design theory. For example, scholars have used agent-based models to simulate and uncover the potential relationships between complex early-stage design processes and design outcomes [13], how parameter estimation heuristics influence design outcomes [14], and how team communication patterns, specialization, and team members' cognitive styles interact to affect design team performance [15]. In the field of innovation management, system dynamics simulations have been used to characterize the drivers of persistent "firefighting" behavior among product development teams [16]. And, in a seminal study in the field of organization studies, March used simulations to characterize the dynamics of exploration versus exploitation behavior in organizations [17].

Across design, management, psychology, and other fields, simulations have proven valuable in understanding how different

[1]Corresponding author.

Contributed by the Design Theory and Methodology Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received October 16, 2025; final manuscript received February 19, 2026; published online March 11, 2026. Assoc. Editor: Christopher Mccomb.

variables might affect outcomes, predicting human behavior across scenarios, understanding collaboration, competition, and decision-making dynamics, among other phenomena [18,19]. Davis et al. proposed that computational simulations can be useful for developing theory and can be employed to understand and further develop existing theories [19]. They argued that simulations are most effective for providing insight into complex theoretical relationships among constructs, developing and testing precise specifications of theoretical logic and assumptions, and revealing emergent outcomes from the interaction of multiple constructs or processes over time. Thus, simulations are well suited to further develop nascent theories that are trajectory based, involving a process that unfolds over time, and theories that follow a known logic structure [19].

In design research, theories such as C-K and Function-Behaviour-Structure (FBS) [20,21] are expressed a structured and trajectory-based form, both verbally and diagrammatically, but they are rarely made executable. Simulating such theories could be useful because (1) it forces informal notions, such as "expand a concept," into precise procedures, exposing where the theory is underspecified, (2) it reveals the emergent dynamics of the theory when iterated over many steps, which are hard to study at scale with human designers, and (3) it stress tests the theory's internal coherence independent of individual designers' unique traits.

Large language models (LLMs) offer attractive new possibilities for simulating and further developing design theories as they can carry out complex reasoning steps directly from natural language descriptions of a theory. Recent studies have focused on exploring how design theories like FBS and Theory of Inventive Problem Solving (TRIZ) can be used to inform and improve the outcomes of LLM-based design tools [22,23]. This study explores a different approach. We move from using design theory to inform the development of LLM-based design tools to using LLMs to simulate and further develop design theory.

The goal of this article is to explore how computationally simulating a design theory can generate new insights and advancements for the theory based on the simulation's findings. In this study, we use C-K theory as our representative case study because (a) the theory provides a clearly defined two-space structure (concepts C and knowledge K) linked by four operators, which lends itself to a trajectory-based computational implementation and (b) C-K theory follows a structured methodology, making it easier to simulate than other design theories [20,24,25].

We developed a simulation using pretrained LLMs. The simulation is designed to follow the C-K theory methodology in both wording and framework. We want to emphasize that the focus of this article is not on C-K theory itself. Rather, our work aims to highlight the utility of simulation in design theory and explore whether there are nuances to consider when using simulations to enhance our understanding of the design theories developed by the community.

2 Background

2.1 C-K Theory. C-K theory delineates two distinct spaces: the concept space (C) and the knowledge space (K) (see Fig. 1) [20,24]. The knowledge space contains propositions with a logical status, meaning elements in this space can be evaluated as true or false [20,24]. Thus, it can be viewed as embodying what is known and accepted by a designer. There is the caveat that the logical status can be based on standard or nonstandard logical systems, but for the purpose of our article, we simplify the logic to the classic true or false logic. As mentioned earlier, the knowledge space is expandable, allowing for the addition of new knowledge over time. However, it is important to note that conflicting views and uncertainties can also be a part of K provided they are considered established knowledge available to the designer [20].

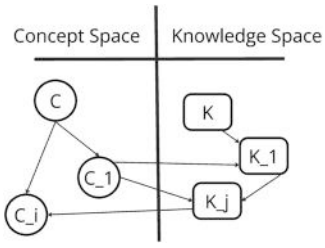

Fig. 1 Representation of C-K theory as proposed by Ref. [24]

On the other hand, the concept space consists of propositions or groups of propositions that lack a logical status in the knowledge space [20,24]. Elements in this space are called concepts, and they are essentially ideas or properties that cannot be proven true or false within the current knowledge space. They serve as the starting point for design by representing new possibilities that are not yet integrated into the existing knowledge space [20,24]. Like the knowledge space, the concept space is also expandable, permitting the generation of new concepts during the design process.

The key distinction appears to lie in how the information is treated [25]. If the information serves as a basis for further exploration and is open to modification or rejection, it likely aligns more with the concept space. However, if the information is accepted as a foundational element, even if the information contains some uncertainty, it likely aligns more with the knowledge space. This approach to distinguishing how C and K are treated has been discussed in the prior literature that talked about the importance of situating the theory around the current environmental context [26].

The process of C-K theory begins with an initial concept C_0, a proposition such as "there exists an entity x with attributes A_0 that is neither true nor false in K" [20]. This concept represents the initial idea. As the design process unfolds, the initial attributes A_0 are modified, leading to new sets of attributes A_i and design parameters D_i. This results in new propositions C_i whereby "there exists an entity x with attributes A_i and design parameters D_i" [20]. Each new proposition C_i is evaluated against the current knowledge space K [20]. Three possible logical outcomes exist for C_i: C_i is false in K, C_i is true in K, or C_i is neither false nor true in K [20]. If C_i is false in K, the design process must alter some attributes or design parameters, indicating the concept will likely require further change [20]. If C_i is true in K and is a candidate for a solution for X, the attributes A_i and design parameters D_i together form a potentially viable solution to the design problem [20]. If C_i is neither true nor false in K, it becomes a new concept and the design process must continue [20].

This logic implies that new concepts are generated as design processes explore possibilities, and new knowledge is added as concepts are tested and validated or invalidated. This process is iterative, with continuous feedback between C and K. Thus, C-K theory defines "design as a reasoning activity that starts with a concept (an undecidable proposition regarding existing knowledge) about a partially unknown object x and an attempt to expand it into other concepts and/or new knowledge" [20]. Among the knowledge generated by this expansion, certain new propositions can be selected as new definitions or new objects arise.

Thus, within this framework, we note that there exist four possible operators for C-K theory: $C \rightarrow C$, $C \rightarrow K$, $K \rightarrow K$, and $K \rightarrow C$ (Fig. 2). Each operator facilitates the expansion of spaces C and K. $C \rightarrow C$ operates to expand concepts by partitioning and exploring new attributes. $C \rightarrow K$ transforms concepts into knowledge by validating propositions (e.g., C_i). $K \rightarrow K$ expands knowledge by adding new validated propositions, and $K \rightarrow C$ generates new concepts from existing knowledge.

We note, however, that there are few explicit guidelines on when to perform one operation over another (e.g., $C \rightarrow C$ versus $C \rightarrow K$ or $K \rightarrow K$ versus $K \rightarrow C$). Thus, we implicitly inferred the guidelines from the prior literature that discussed the industrial

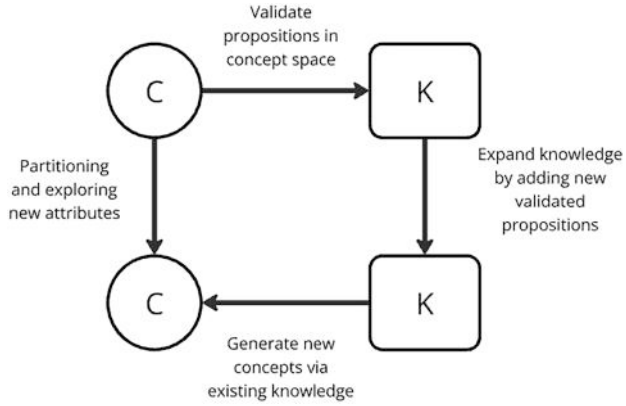

Fig. 2 The set of four possible operators for C-K theory, collectively known as the design square in the original literature [24]

application of C-K theory [25]. Therefore, if one is deciding between $C \rightarrow K$ versus $C \rightarrow C$, one should perform $C \rightarrow K$ when the concept can be compared against existing parameters or knowledge to decide whether it can be validated or integrated within the existing knowledge [25]. If the concept is still ambiguous or its potential remains unexplored within the current knowledge base, perform a $C \rightarrow C$ operation. Similarly, if one is deciding between $K \rightarrow C$ versus $K \rightarrow K$, $K \rightarrow K$ is done when the knowledge base itself can be expanded through new validated propositions or insights [25]. If the current knowledge instead suggests potential new concepts, a $K \rightarrow C$ operation would be conducted to generate those concepts.

2.2 Large Language Models. LLMs are machine learning models capable of reasoning and generating creative output through natural language generation [27]. These models are primarily trained on large text datasets to predict subsequent words or sequences in a sentence, which, at the time of writing this article, are trained primarily based on transformer architectures that enable efficient training on large datasets [28]. These models are able to decipher patterns, nuances, and context within the human language, and recently, there has been an increased development of more tailored pretrained language models that have increased capabilities in creative reasoning (e.g., GPT-4.5) [29].

With these increased capabilities in LLMs, they have been applied across a range of design tasks, including design concept generation, simulating artificial human empathy, and specific design processes such as TRIZ and FBS [22,23,30–32]. They have also been utilized within the framework of C-K theory, where Chen et al. established a human–AI approach in which the LLM assists designers in retrieving knowledge and uncovering new concepts using C-K theory as a guideline [33]. Thus, we note the potential capability of LLMs to perform design reasoning activities, rendering them valuable tools for simulating C-K theory.

At the same time, the recent design theory work has begun to characterize GenAI's generative profile using C-K theory. Bordas et al. show that current GenAI models, including LLMs, mainly exhibit *design-by-activation* within a fixed representation; thus, they generate by reactivating and recombining existing knowledge rather than restructuring the underlying knowledge base [34]. In contrast, Le Masson et al. show from the Bauhaus case that strong or "generic" generativity in design is associated with *reordering* or *splitting* knowledge bases [35]. Thus, we treat LLMs in this article as constrained C-K agents and interpret our results with these limitations in mind.

2.3 Using Computational Simulation to Build Theory. One of the earliest applications of simulations for theory development was Norbert Wiener's work in cybernetics [36]. Wiener proposed that systems, whether mechanical or biological, primarily function through feedback processes. He observed that systems self-regulate via feedback and suggested that human behavior could similarly be understood in terms of input and output. He argued that cybernetics theory could help further our understanding of human behavior through the principles of feedback, regulation, and information exchange. Later on, Simon and Newell explored theories related to problem solving [9]. They proposed a theory of problem solving whereby a problem space comprises an initial state, a goal state, and all possible intermediate states of a problem. They then relied on heuristics to guide the navigation of this problem space. By using computational simulations, they developed and tested theories about human problem solving to understand how different strategies and heuristics might interact and function. This work was broadly influenced by Simon's earlier work, *The Sciences of the Artificial*, where he argued that design is central to all human-made constructs, with the core goal being to create solutions that meet specific objectives within given constraints [10].

These examples illustrate that computational simulation can be a significant methodological approach to theory development. However, using simulations to build theory has also faced criticisms, with some noting that simulations are just toy models that either replicate the obvious or are so detached from realism that they fail to yield any useful and real-life applicable results [37]. Such criticisms often focus on the underlying assumptions that may operate within a simulation, which are frequently noted to be unrealistic. In a work by Davis et al., the authors argue that simulation can, instead, be valuable as the intermediate step between theory building and theory testing [19]. They argue that simulation can help develop nascent or interim theories into more precise and comprehensive ones and is particularly useful for studying processes over time. In our work, we build a simulation of C-K theory. This theory has a formal logic and involves a process that operates over time, which can be difficult to capture in real-world design data. We use C-K theory as a case study for using LLM agents to simulate a design theory in attempts to further understand or explicate that theory and potentially offer directions for further refinement.

3 The Present Research

We conducted three studies using a mixed-methods approach. Study 1 implements an initial C-K simulation that chooses among the four operators based on a strict interpretation of the theory [24] and examines what transition patterns and diversity dynamics emerge. Motivated by the dominance of a $C \rightarrow C$ loops and the decline in generated concept diversity in Study 1, Study 2 keeps the simulation architecture fixed but varies the initial knowledge space to test how different configurations of K shape C-K trajectories and diversity. Study 3 then turns from "what happens" to "why it happens" by qualitatively analyzing the LLM-generated rationale statements that accompany $K \rightarrow C$ transitions in order to characterize the type of reasoning the agent uses when moving from knowledge back to concepts. Taken together, the three studies move from observing the emergent behavior of the C-K simulation (study 1), to probing the influence of the knowledge space structure (study 2), to uncovering the internal logic of a critical operator (study 3).

4 Study 1: Simulating C-K Transitions Using Large Language Models Agents With a $C \rightarrow C$ Penalizer

4.1 Constructing the Simulation. We encoded C-K theory into an LLM agent, attempting to replicate the logic and operations of C-K theory as accurately as possible according to the literature [20,24]. The simulation follows the structure as shown in Fig. 3. The agent takes in the concept space and the knowledge space to decide among four possible actions: $C \rightarrow K$, $C \rightarrow C$, $K \rightarrow C$, $K \rightarrow K$. Note that the agent can only perform a $C \rightarrow C$ or $C \rightarrow K$

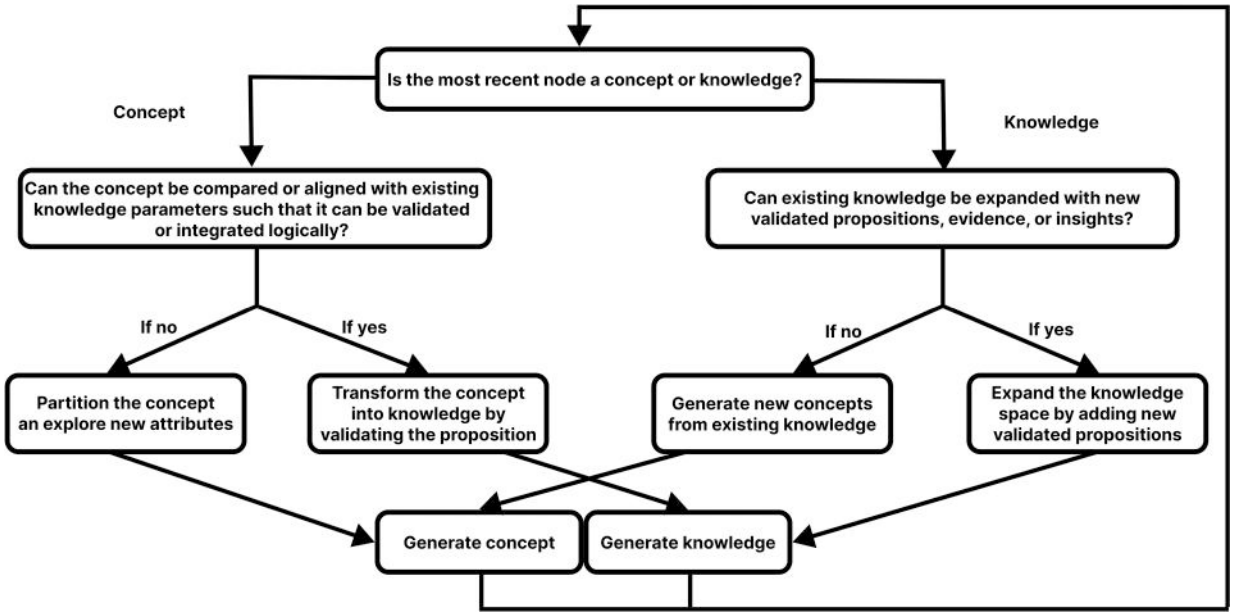

Fig. 3 Overall framework of the C-K theory simulation logic

operation when the current element fed into the agent is a concept, and similarly, $K \rightarrow K$ or $K \rightarrow C$ when the current element is knowledge.

The design topic we chose was one taken from the prior literature: "design a creative nail holder for use while hammering a nail" [25]. We initialized the simulation by constructing a knowledge space for our design topic. We developed this knowledge space by generating design requirements using GPT-4.5 [29] and validating them with design experts experienced in product design. We included 20 design parameters in the knowledge space. The knowledge space we used is available in our GitHub repository.[2]

The first primary component of this simulation model involves deciding whether to go to $C \rightarrow C$ or $C \rightarrow K$ when in a concept state or $K \rightarrow K$ or $K \rightarrow C$ when in a knowledge state. The second primary component is the operation that the LLM performs when it is at $C \rightarrow C$, $C \rightarrow K$, $K \rightarrow K$, or $K \rightarrow C$. To simplify our simulation, we chose to have the agent generate the output in a *Concept/Knowledge Title:* and *Concept/Knowledge Description:* format.

The first decision-making logic that the agent must perform is to determine, when in a concept space, whether to proceed with concept or knowledge. The function takes in the current concept title, concept description, past operation transformations (which are updated in each iteration), and the existing knowledge base. The agent then decides whether to perform a $C \rightarrow C$ or a $C \rightarrow K$ operator based on the following logic: choose $C \rightarrow C$ if the concept is ambiguous or novel and needs further exploration, or $C \rightarrow K$ if the concept is refined, aligns with existing knowledge, or has the ability to enrich the knowledge base.

Similarly, when in the knowledge space, the decision to proceed with concept or knowledge is based on the literature as described in Sec. 2. Here, the function takes in the current knowledge title, knowledge description, past transformations, and the existing knowledge space. The agent then decides whether to perform a $K \rightarrow C$ or $K \rightarrow K$ operator based on the following logic: if existing knowledge suggests new concepts or insights, initiate $K \rightarrow C$ for concept expansion; if new information is validated and can be integrated into the knowledge space, initiate $K \rightarrow K$ for knowledge expansion.

[2]https://github.com/kevinma1515/simulation_jmd

The agent will then execute one of the four operations after the agent has decided whether to conduct either $C \rightarrow C$, $C \rightarrow K$, $K \rightarrow K$, or $K \rightarrow C$. When the agent executes the operation, the agent will consider the context of the existing knowledge space, the current concept description and title, and past transitions. However, the degree in which this occurs depends on the experimental conditions. As referenced in Sec. 2, the agent was developed to execute each operation according to the C-K theory literature: $K \rightarrow C$ is to use existing knowledge to generate new concepts, $C \rightarrow K$ is to validate or transform the concepts into knowledge, $C \rightarrow C$ is to expand or modify the concepts within the conceptual space, and $K \rightarrow K$ is to refine or expand the existing knowledge.

In our implementation, the agent follows a reactive decision policy, relying only on the currently active concept or knowledge item to select between the different operators. This collapses the C-tree into a single active branch and does not allow the agent to revisit or prune earlier concepts. Similarly, K is only ever expanded by appending new propositions, so existing knowledge is never revised or reordered. As such, according to Hatchuel et al.'s derivation of the C-K theory in their later works, we only model expansion of C and K but not K-reordering or C-tree reordering [38]. Operationally, we also approximate concepts as propositions that appear novel relative to the knowledge space rather than as formally undecidable statements in a strong C-K theory stance. The simulation should therefore be treated as an LLM acting as a constrained C-K agent under these simplifications, not as a full implementation of C-K's formal notion of undecidability. The consequences of these choices will be discussed in Sec. 4.4.

4.2 Experimental Conditions. During preliminary simulations in which the agent operated solely on the concept and knowledge spaces, we observed that it overwhelmingly selected the $C \rightarrow C$ operation, producing long sequences of concept expansion operations with very few $C \rightarrow K$ transitions. To address this imbalance, we introduced a penalizer (also referred to as with/without penalizer in Sec. 4.4) designed to discourage repetitive concept generation and to encourage transitions into the knowledge space. When the penalizer is activated, an additional LLM layer evaluates the novelty and potential of the most recent concepts. If the agent repeats similar ideas across five consecutive iterations, the penalizer intervenes by forcing the next operation to be $C \rightarrow K$. This penalizer acts as a feedback control that forces knowledge

space entry whenever the concept expansions lead to stagnant concepts (e.g., repeated $C \rightarrow C$ operations that yield only minor textual variations around the same underlying idea). The penalizer is therefore a non-C-K feedback control that we add on top of the simulation to counter this LLM-specific tendency and to mimic similar defixation strategies documented for human designers and GenAI [39,40]. When the penalizer is deactivated, the simulation does not use this intervention. The prompt logic used for the penalizer is provided in Appendix A.

4.3 Computational Evaluation. In each iteration, we generated a concept comprising various concept titles and descriptions. We converted these titles and descriptions into embeddings using OpenAI's embedding model named *text-embedding-3-small* and set the embedding dimension to 512. These embeddings were then used to measure computational diversity across each time-step iteration. To measure diversity, we used both determinantal point process (DPP) and the average distance to the centroid. This method of calculating diversity has been used in prior works [31,41].

The DPP framework we used to calculate diversity was based on the properties of the eigenvalues of the cosine similarity matrix. All embeddings were first normalized so that the cosine similarity between any two vectors lies in the range $[-1, 1]$. Since DPPs require a positive semi-definite kernel matrix, the cosine similarity matrix was transformed to nonnegative values, and the resultant matrix was used to compute its eigenvalues. We then logarithmically scaled the eigenvalues and took the average of all the values. Therefore, a more negative output from our DPP framework was due to a smaller eigenvalue, which implies that a more negative DPP output in our graph indicates a less diverse embedding space.

The average distance to centroid framework we used assesses how the embeddings are distributed around the centroid at each time-step. We calculated average distance to centroid by computing the mean distance of all embeddings from the centroid. Thus, a smaller average distance indicates that the points are closely clustered, while a larger average distance would suggest that the points are more spread out. We had to reduce the dimensionality of the text embeddings to nine dimensions to make the computation of the average distance to centroid feasible. The complete algorithm and code for this process are available in our GitHub repository.

4.4 Results and Discussion. Our simulation investigated how LLM agents enact C-K theory operations under two conditions: the presence versus the absence of a penalizer feedback loop and the use of two different model types (*gpt-4o* and *gpt-4.5*). The penalizer was designed to discourage repetitive concept generation by forcing a $C \rightarrow K$ transition after five consecutive similar outputs. Across conditions, we evaluated transition histories and concept space diversity over time, with results shown in Fig. 4. This combined results and discussion section reports the empirical findings alongside the interpretations to connect the simulation behavior with implications for C-K theory.

4.4.1 Transition History. Across all model-penalizer combinations as shown in Fig. 4, the $C \rightarrow C$ operation was by far the most dominant transition type. In the without penalizer condition, $C \rightarrow K$ transitions were almost entirely absent, and the simulation got stuck in prolonged sequences of concept expansion. When the penalizer feedback loop was activated, $C \rightarrow K$ transitions appeared consistently after approximately five time-steps, the point at which the penalizer intervened because the prior concept generations were judged to be repetitive.

From a theoretical view, C-K does not prescribe a fixed rate of operator use, but it does emphasize a continuous interplay between C- and K- expansion and the possibility to return to K when conceptual exploration stalls [20]. Operationally, we expected bursts of $C \rightarrow C$ exploration with occasional $C \rightarrow K$ operations once a concept had been elaborated enough to be tested against K or began to resemble existing knowledge. Instead, even in such situations, the agent almost always continued with $C \rightarrow C$. This suggests that, when given a choice, the LLM tends to treat design as an open-ended text elaboration task that adds more details to the current description rather than as a search process that periodically returns to K to test, reorganize, or generalize concepts, which is a tendency that is further amplified by our single-branch implementation.

4.4.2 Diversity Evaluation. We then assessed how concept diversity evolved over time as shown in Fig. 4 using two metrics: DPP and average distance to centroid, as described in Sec. 4.3. In this case, more negative values for DPP indicate lower diversity and smaller values for average distance to centroid indicates tighter clustering.

Across all conditions, both measures showed declining diversity as iterations progressed. DPP values became more negative, and the average distance to centroid decreased over the time-steps, suggesting that newly generated concepts grew increasingly similar. In addition, our initial expectation was that forcing $C \rightarrow K$ transitions through the penalizer or using a more capable reasoning model (e.g., *gpt-4.5*) might sustain or increase diversity. However, diversity declined in all conditions. Although the feedback loop successfully triggered transitions into the knowledge space, it did not prevent convergence toward repetitive cluster of generated concepts.

That being said, under a richer implementation that includes K-reordering and explicit splitting of knowledge structure [35,38], occasional renewals of diversity would be expected to

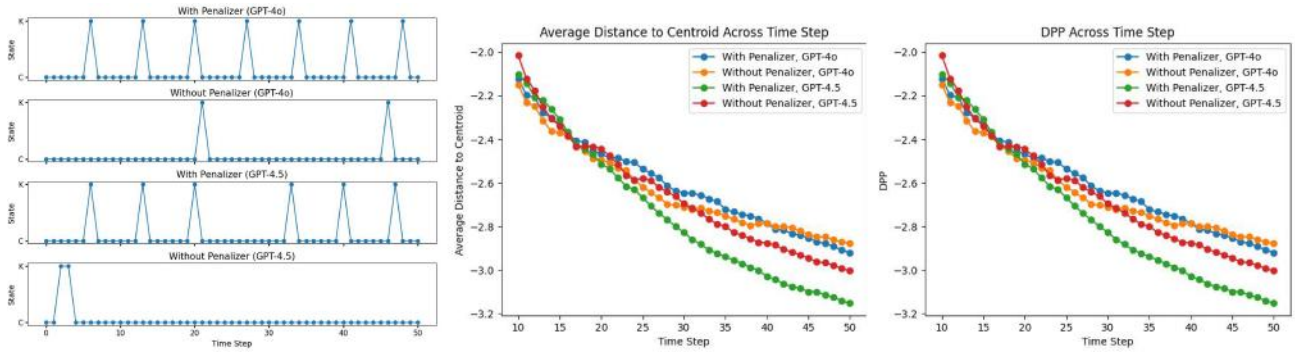

Fig. 4 Transition history and concept diversity results for study 1. On the left are the transition history. This shows, for each condition, which space the agent occupies at each time-step. Without the penalizer, the trajectories quickly collapse into long runs of C-space with rare visits to the K-space, whereas with the penalizer, the state oscillates regularly between C and K, roughly every five steps. The right-hand panels plot concept space diversity over time using average distance to centroid and DPP. In all conditions, the curves drift downward.

occur as the agent can bring structurally different knowledge into the concept space. Our simplification gives the agent no such mechanism as K is only expanded and never reorganized, and $K \rightarrow C$ is specified only in very general terms. The observed diversity decline, therefore, reflects a combination of known convergence and fixation phenomena in LLM-generated content, e.g., Ref. [42] and the limitations of our current simulation, rather than strict evidence that a C-K-based approach itself leads to shrinking concept diversity.

4.4.3 Qualitative Illustration. A representative run from the *gpt-4o* model with the penalizer enabled illustrates this pattern in Table 1. Around time-step 23, the simulation began generating "quantum"-related concepts. Yet, by time-step 37, it had returned to "quantum" themes even though a $C \rightarrow K$ and $K \rightarrow C$ operation occurred in between due to the penalizer feedback loop.

This recurrence happened despite the forced transition occurring between those time-steps and confirms that, rather than expanding into new conceptual territory, the system often reverted to thematic attractors inherent in the model's generation pattern. These attractors limited sustained exploration of the concept space, leading to a potential explanation behind the overall decline in measured diversity.

4.4.4 Understanding Concept to Concept Loops: Addressing Sparse $C \rightarrow K$ Transitions. Findings from this study reveal that the $C \rightarrow C$ operation was by far the most dominant operation, while the $C \rightarrow K$ operation occurred only very rarely. We suspect that this issue stems from the fact that once the simulation performs a $C \rightarrow C$ operation, it encourages the addition of novel partitions to the concept. Thus, the underlying idea of each concept remains the same across each time-step, but with each $C \rightarrow C$ operation, new elements are added to the concept. This has the downside of making the concept more challenging to validate with the existing knowledge space, as it becomes increasingly impractical and disconnected from the established work with each time-step.

Even though we tried our best to follow the literature as strictly as possible, this could be an issue with the way we interpreted the decision logic between doing $C \rightarrow C$ versus $C \rightarrow K$. One possible issue is that the LLM agent may require different wording in its prompt that is not as exact as the wording from the design literature to get the simulation to perform better. More fundamentally, our results highlight that the textual descriptions of C-K transitions do not, on its own, function well as a decision policy. Additional structures, such as explicit rules for revisiting earlier concepts, reordering the concept tree, or splitting, and restructuring K into new knowledge bases [35,38] will be necessary if one wants to turn C-K theory into a simulation.

4.4.5 Addressing Decreasing Concept Diversity: It Matters Not Just Whether and When But How $K \rightarrow C$ Transitions Occur. Across all conditions, the diversity values trended downward. This raises the question: If new concepts are generated through the $K \rightarrow C$ operation, why are they not leading to higher diversity? The issue is that the LLM agent appears to always revert to previous ideas (see Table 1 for an example). This suggests that merely performing a $K \rightarrow C$ operation may not suffice for generating innovative and creative ideas, indicating a potential need to revisit what the $K \rightarrow C$ operation entails in C-K theory. This implies that merely transitioning into the knowledge space and exiting it is not what matters, but rather the manner in which this transition is executed is what is more important.

Overall, our findings indicate that the logic for $K \rightarrow C$ transitions in C-K theory is not sufficiently specified in our implementation to ensure the generation of diverse or innovative concepts. This is consistent with the prior work showing that both humans and GenAI tend to remain trapped in a small number of solution categories unless knowledge is deliberately split or reorganized [35,39,40]. Prior works have shown that what matters for better generated concepts is not simply performing a $K \rightarrow C$ transition but rather which knowledge structures currently exist and how they are reconfigured in the concept space [35,39,40]. Thus, the declining diversity we observed should not be seen as evidence that C-K theory itself is nongenerative, but rather showing that, without explicit mechanisms to explore the knowledge space, an LLM-driven C-K theory agent tends to produce fixation patterns. This points to areas where more operational transition rules and knowledge-structuring mechanisms are needed if C-K theory is to be used as an algorithmic design engine and suggests potential connections with the existing work on fixation, inspirational stimuli, and design by analogy [43–48]. Motivated by this, study 2, in Sec. 5, takes a preliminary, minimal step toward richer knowledge structuring by varying the initial knowledge space. This serves as a diagnostic of how far simple K-priming can go in an LLM-driven C-K theory simulation.

5 Study 2: Simulating C-K Transitions Using Large Language Model Agents With Different Knowledge Bases

5.1 Constructing the Simulation. Methods in study 2 follow the same agent architecture, operation logic, and prompting structure described in study 1 as shown in Sec. 4.1 and Fig. 3. As mentioned earlier, the agent iteratively selects among the four C-K operators $C \rightarrow C$, $C \rightarrow K$, $K \rightarrow C$, and $K \rightarrow K$ to simulate theory-driven transitions within the design task "design a creative nail holder for use while hammering a nail." The main modification introduced in this study was a dictionary-based data structure for all simulation elements. Each concept or knowledge instance was encoded as a record with four fields: type identifier (e.g., C_i or K_i), title, description, and operation rationale, which are all updated at each iteration. The operation rationale captured the agent's explanation for why a particular transition was chosen and was fed back into subsequent decision steps. This schema improved traceability of reasoning chains and consistency across operations.

Unlike study 1, where the initial knowledge space was fixed, in study 2 the composition of K varied by experimental conditions as we will discuss in Sec. 5.2. Note, the penalizer feedback mechanism from study 1 was not used in study 2 in order for us to isolate and understand the effects of knowledge space initialization.

Table 1 Example concept or knowledge: *with penalizer feedback—GPT4o condition*

ID	Type	Title of the concept
23	C	Quantum Resonance Nail Holder With Integrated Environmental Adjustments
24	C	Bio-Mimetic Nail Holder With Integrated Environmental Adjustments
25	C	Advanced Neuro-Adaptive Nail Holder With Contextualized AI Interactions
26	C	Bio-Symbiotic Intelligent Nail Holder With Reactive Material Sensing
34	C	Dynamic Tactile Feedback Nail Holder: Enhancing Safety and Precision With Adaptive Material Technology
35	C	Neuro-Haptic Adaptive Nail Holder With Enhanced Environmental Responsiveness
36	C	Cognitive-Assisted Symbiotic Nail Holder with Ecosystem Integration
37	C	Quantum-Enhanced Cognitive Nail Holder With Dynamic Ecosystem Interaction

5.2 Experimental Conditions. For our experimental setup, we employed *GPT-5* as the LLM for our simulation. GPT-5 was selected primarily because it represents a state-of-the-art model capable of sustaining extended reasoning chains and maintaining contextual coherence, both of which are necessary for implementing a computational representation of the C-K theory operations [49]. At the time we ran study 2, the GPT-4.5 model used in study 1 was no longer available through the provider's API, so we used the most recent successor model. Because the objective of this study is not to compare model architectures but rather to examine whether a contemporary LLM instantiates C-K theory through simulation, we treat the LLM as a black-box reasoning model and do not interpret differences between studies as model comparison effects.

The simulation was executed under varying initial conditions to investigate the effect of knowledge space priming on the simulation. The rationale behind varying knowledge space initialization is that, as described in the C-K framework, the structure and content of the knowledge space (K) strongly influence the subsequent paths of concept expansion (C) [24]. By altering the initial K-state, we aimed to determine how different knowledge contexts might lead to different C-K pathways, operational patterns, and textual outputs.

As such, we developed three primary experimental conditions reflecting distinct initial knowledge space configurations:

- *Human-Centered Design Knowledge Priming*. In this condition, the initial knowledge space was populated with propositions and best practices drawn from the human-centered design process. Human-centered design is a widely practiced design process and has been well-studied in many contexts [50]. As a result, we selected this configuration to provide a test case for whether a human-centered design-oriented K-space promotes broader conceptual expansion within the simulation. The specific knowledge entries used here are provided in Appendix B.
- *No Prior Knowledge.* In this second condition, the knowledge space was left initially empty. This served as a counterfactual baseline to evaluate how an unconditioned agent behaves without preexisting knowledge constraints. By removing priors, this condition allows us to observe the model's intrinsic design reasoning pattern and assess the degree to which external knowledge structure shapes exploration in C-K terms.
- *System Requirements Priming.* The third condition initializes the knowledge base with system-oriented requirements commonly seen in systems engineering practices [51]. In general, this configuration should emphasize convergent reasoning as, in systems engineering, the goal is constraint satisfaction as opposed to the divergence associated with human-centered design. The intent here was to test whether a more structured or constraint-heavy initial K-space would lead to narrower conceptual elaborations or more rapid convergence to validated knowledge propositions. The specific system requirement entries are provided in Appendix C.

Each simulation was executed independently five times per condition, resulting in 15 independent simulation runs. The decision to replicate each condition was based on the stochastic nature of LLM outputs because each run introduces inherent variability due to probabilistic token sampling. Replicating runs enables us to examine whether the observed conceptual progressions are robust to stochastic variations and whether consistent high-level behavioral patterns emerge across independent runs of the same setup.

5.2 Exploratory Temporal Replication. To test the temporal stability of LLM behavior (e.g., LLM models may drift as the product gets updated periodically by the creator of the model), we repeated the human-centered design knowledge priming condition 1 week apart from the initial runs using the same inputs, prompts, and codebase. Because this analysis is primarily exploratory and complementary to our qualitative work, we present the results in Appendix D.

5.3 Computational Evaluation. Computational analysis procedures were identical to those in study 1 as shown in Sec. 4.3. Concept titles and descriptions were embedded using OpenAI's *text-embedding-3-small* and diversity was quantified with the same two metrics: DPP and average distance to centroid (after 9-dimensional reduction). All normalization, kernel, and eigenvalue computations followed the same implementation as previously described, and the codebase is available in the GitHub repository.

To complement the quantitative diversity metrics, study 2 incorporated a text-analytic procedure using the *spaCy* library [52]. Structured JSON outputs containing all concept and knowledge records were processed to extract frequent trigrams within each simulation and were aggregated across different runs. These phrase frequency profiles characterized thematic differences among the knowledge priming conditions and supported qualitative interpretation of simulation outputs [53].

5.4 Results and Discussion

5.4.1 Simulation Behavior Across Knowledge Priming Conditions. Across all the knowledge-priming conditions, as shown in Fig. 5, the simulation exhibited broadly similar temporal patterns in the alternation between concept and knowledge spaces. Each condition began with an initial phase defined by their respective preset concept and knowledge states (if any) before stabilizing into longer sequences dominated by concept state operations. Although the rate of transitions and the duration of concept-dominant phases varied slightly between conditions, the overall temporal structure remained consistent.

In the *Human-Centered Design Knowledge Priming* condition, each simulation began from the preset initial concept and knowledge states. Following this initialization, the simulation stabilized into extended periods of concept expansion. Occasional returns to the knowledge space occurred throughout later iterations, but the general pattern was a sustained $C \rightarrow C$ transitions. All five runs displayed similar sequences with minimal variability.

The *No Prior Knowledge* condition followed a comparable trajectory. The simulation contained predominantly $C \rightarrow C$ transitions with the occasional transitions to and out of the knowledge space. This was consistent across all runs, and the overall progression mirrored the structure observed in the human-centered design condition.

In the *Systems Requirement Priming* condition, the simulation again contained broadly similar temporal pattern to the other two conditions. Some runs exhibited slightly earlier stabilization than in the other two conditions, but the general pattern of early alternation followed by predominantly $C \rightarrow C$ transitions was preserved. Minor differences in transition timing were observed across replicates; however, the overall temporal patterns were consistent.

5.4.2 Quantitative Diversity Analysis. To quantify the evolution of conceptual diversity over time, two independent measures were computed: average distance to centroid and DPP. In all three conditions, both average distance to centroid and DPP scores showed a monotonic decrease over time, indicating a consistent reduction in conceptual diversity as the simulations progressed. The overall shapes of the curves were similar across conditions, which consisted of an initial steep decline followed by a more gradual decrease in DPP or average distance to the centroid values. Minor variations were observed in the rate of early decline across different conditions. Replicate simulations within each condition produced nearly parallel trajectories, which demonstrates that model behavior remained relatively stable between the replicates despite inherent stochasticity in LLM models.

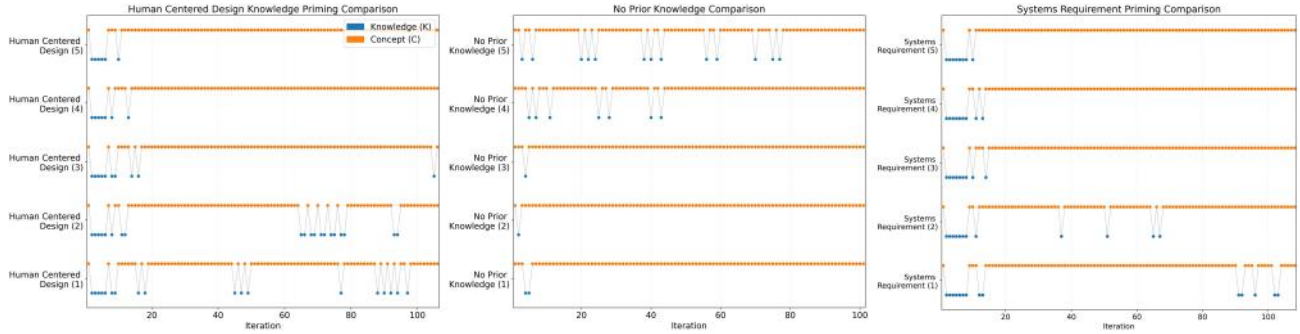

Fig. 5 C-K transition patterns across different knowledge priming conditions in study 2

5.4.3 Phrase Frequency. A trigram frequency analysis was conducted to examine the linguistic patterns across the textual outputs of all simulation conditions. While several trigrams were shared across conditions, the results primarily revealed a consistent procedural language reflecting the simulation's internal implementation of C-K theory. For example, across all conditions, phrases such as "remain undecidable current knowledge," "validate existing knowledge," and "concept step appropriate" appeared repeatedly. These expressions appear to originate from the structured prompts guiding the agent's reasoning and correspond to the four C-K operators.

Beyond these procedural expressions, moderate qualitative differences emerged in the conceptual content of generated ideas. In the *Human-Centered Design Knowledge Priming* condition, the agent produced concept labels emphasizing user interaction, adaptability, and co-creative or "smart" functionality (e.g., *Cognitive-Assisted Nail Holding System* and *Predictive Co-Cognitive Nail Holder*). The *No Prior Knowledge* condition, by contrast, generated object-centered descriptions that iteratively modified physical features of the nail holder (e.g., *Smart Magnetic Nail Holder* and *Predictive Morphing Nail Holder*). Finally, the *Systems Requirement Priming* condition produced language that highlighted performance and controllability (e.g., *Magnetic-Field Nail Holder* and *Neuro-Rhythmic Harmonic Holder*). These differences suggest that while the simulation's operational rationale remained quite consistent, the knowledge space initialization influenced the thematic framing of the conceptual expansion.

5.4.4 Discussion of Findings. Across all conditions, the agent exhibited a strong tendency toward repeated $C \rightarrow C$ operations, indicating that the concept expansion was the dominant behavioral mode regardless of initial knowledge framing. This replicates the pattern observed in study 1. The fact that different K-primings do not change the underlying dynamics is consistent with C-K theory's core claim that concept trajectories are shaped by the current knowledge base [24].

However, this continual concept expansion did not translate into broader exploration as diversity decreased across all conditions as shown in Fig. 6. Even when the knowledge space was primed with human-centered or system-oriented knowledge, conceptual exploration converged over time toward familiar or self-similar conceptual clusters. In theory, $K \rightarrow C$ transitions should introduce new directions by drawing on available knowledge to stimulate novel concepts; however, in practice, these transitions appear to have been limited to recombining semantically close information rather than generating "far" analogical or disruptive ideas [48]. This pattern echoes prior findings on fixation and restrictive heuristics in human designers [40] and on GenAI's tendency to deepen existing categories rather than split knowledge bases [35,39].

Thus, what our simulations highlight is not a theoretical flaw in C-K theory, but a practical gap when one tries to operationalize C-K theory as an algorithmic agent. The verbal descriptions of $C \rightarrow K$ and $K \rightarrow C$ do not determine when to leave a $C \rightarrow C$ loop, which parts of K to mobilize, or how to reorder or split K once new concepts appear. Human designers have been found to handle these decisions through tacit heuristics, utilizing the practice of splitting knowledge, and explicit defixation techniques [35,39,40]. In our LLM-only setting, those implicit mechanisms are absent, so the underspecification becomes visible as convergence and fixation. Our experiments do not yet provide definitive prescriptions for these decision rules, but they identify where additional operational detail is needed if C-K theory is to be simulated as a descriptive representation of design.

Taken together, these patterns clarify what happens during simulation but not why. In particular, the mechanisms behind movement between knowledge and concept spaces remain opaque. To probe this further, study 3 turns to a qualitative analysis of the rationale statements generated by the LLM agents when they perform $K \rightarrow C$ transitions.

6 Study 3: Analyzing Rationale Statements on Why Large Language Model Agents Performed $K \rightarrow$C Transitions

The consistent decline in concept diversity across time-steps as observed in studies 1 and 2 raises a key question: why does the simulation move from knowledge to concept space, and how might those moves spark new ideas? In study 3, we build preliminary theory about how and why the simulations perform $K \rightarrow C$ transitions by analyzing the simulation's generated rationale statements. Our goal is to identify the reasoning patterns that drive these transitions—mechanisms that should remain meaningful even if exact simulation trajectory varies across different runs.

6.1 Qualitative Coding Methodology. By using a conventional content analysis approach [54], we qualitatively coded the LLM agents' rationale for why transitions from $K \rightarrow$ C occurred. Our unit of analysis was each qualitative rationale statement generated by the LLM each time it made a $K \rightarrow$ C transition. We assembled a dataset of 62 $K \rightarrow$ C transition rationale statements across five runs of the GPT-5 LLM agent with a design process knowledge base. In some ways, analyzing these five runs of LLM-generated rationale is analogous to analyzing think-aloud data from five designers and design process experts as they complete a design ideation task. Each rationale statement consisted of roughly 75 words, for example:

> The next step should be $K \rightarrow C$ *(Knowledge to Concept)* because recent knowledge has already validated feasibility and established ergonomic constraints, shifting ambiguity from "can it work?," to "what new forms can it take?" This knowledge now acts as a *creative springboard*, enabling the generation of new, concrete design directions rather than requiring further validation. Staying in $K \rightarrow$K would risk stagnation, while $K \rightarrow$C keeps the process innovative and forward-moving.

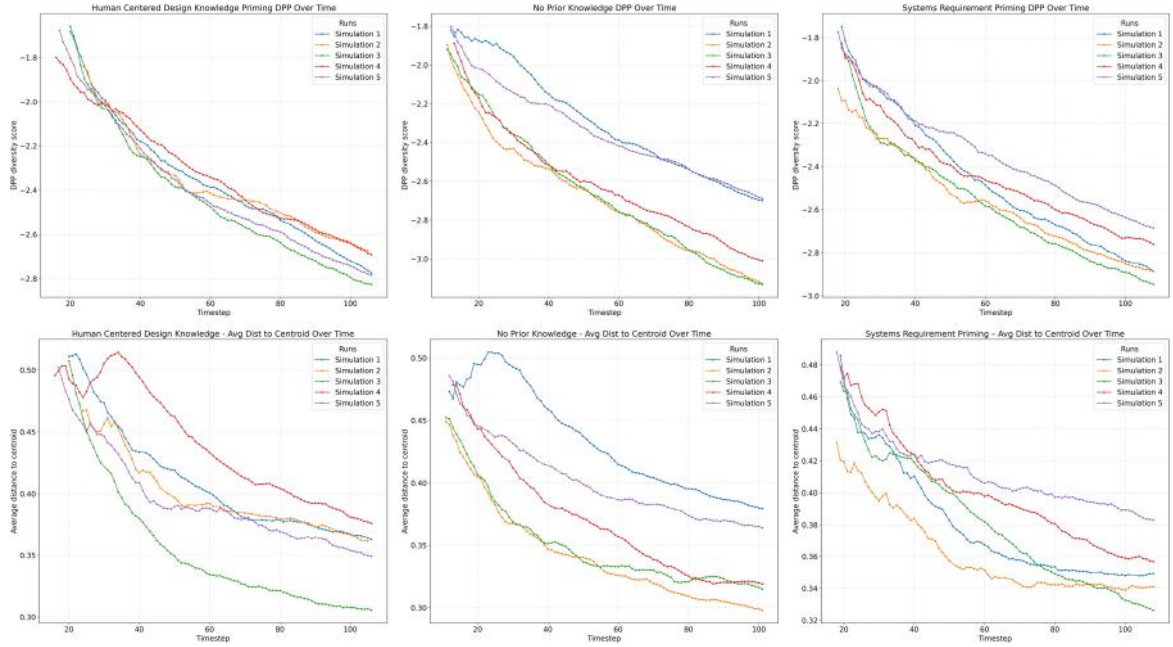

Fig. 6 Diversity over time across knowledge priming conditions in study 2

We analyzed the dataset of rationale statements using conventional content analysis [54,55], starting with incident-by-incident open coding [56] to uncover the major themes in why our LLM agents made $K \rightarrow$C transitions. Our initial codebook was developed through open coding of a subset of the rationale statements [54,55,57] with initial codes like *knowledge de-risks feasibility and triggers more design exploration*, *knowledge shows concept not suitable for all contexts*, *knowledge reveals contradictions that suggest new design directions*, etc. We recoded the open codes until a stable set of primary and secondary codes emerged. The resulting codebook had three primary codes: *knowledge acting as a creative springboard, concept lacks definition,* and *C-K space metacognition*, with three secondary codes under knowledge acting as a creative springboard: *knowledge space reveals design gaps, recombination, or analogies that suggest new design directions*, *knowledge space reveals design trade-offs or contradictions that suggest new design directions,* and *knowledge space reveals design feasibility or other "ility" assessments that suggest new design directions*. By using the refined codebook, we conducted focused coding [54,55,57] of the full analysis dataset (62 instances) until it was clear that saturation had been reached. The second author conducted most of the coding, and both the first and second authors coded a random subsample to clarify the codebook, achieving a Cohen's kappa intercoder reliability score of 0.61, indicating substantial agreement [58]. The final codebook is presented in the following section.

6.2 Results: How Large Language Model Agents Rationalized $K \rightarrow C$ Transitions. Our content analysis yielded five main themes of LLM-generated rationale describing why transitions from knowledge to concept space occurred. These are described in Table 2 and illustrated with examples in the following sections.

6.2.1 Knowledge Acts as a Creative Springboard. The first three themes of self-described rationale of when LLM agents moved from $K \rightarrow$C all involved knowledge, in various ways, acting as a creative springboard leading to new possible design directions.

Knowledge Reveals Design Gaps, Recombinations, and Analogies. The first way that LLM agents described knowledge acting as a creative springboard was by revealing design gaps, recombinations, or analogies that opened new design possibilities. For example, one rationale statement described how "further $K \rightarrow$K would only refine what is already known without generating novelty. Instead, the structured knowledge base now reveals clear gaps, hybrid opportunities, and migration pathways that can only be explored through new concepts." Building upon the notion of clear gaps and hybrid opportunities being revealing, other rationale statements noted that some knowledge can structure a design space into a "taxonomy [that] is generative, opening pathways for new concepts (such as reusable add-ons, non-magnetic integrated mechanisms, or hybrid solutions)." Other $K \rightarrow$ C transition rationale described how the knowledge base had suggested possibilities to "recombine proven mechanisms (e.g., biomimicry, self-healing, energy harvesting, sensory feedback) into novel, unexplored concepts that push the design frontier." In addition to identifying design gaps or recombination possibilities, the knowledge base could suggest near or far analogies that could also spark $K \rightarrow$ C transitions.

Knowledge Reveals Design Trade-offs and Contradictions. Another way that LLM agents described the knowledge space acting as a creative springboard was by revealing design trade-offs or contradictions. For example, an LLM agent described how its knowledge base provided "trade-off insights that reveal unexplored opportunities rather than just factual validations [that] naturally open new design directions, enabling [. . .] concepts that balance safety, ergonomics, cost, and sustainability." In another example rationale statement, an LLM agent shared that:

> The next step should be $K \rightarrow$C because the knowledge space has reached closure with a validated taxonomy of nail holder mechanisms (magnetic, adaptive, wearable), but this also exposes fundamental contradictions such as simplicity versus universality and protection versus usability [that] cannot be resolved through further knowledge testing ($K \rightarrow$ K). [. . .] Instead, they demand the generation of new concepts that transcend the current taxonomy and open unexplored solution spaces.

These $K \rightarrow$ C rationale statements illustrate how knowledge can illuminate trade-offs and contradictions and, in doing so, suggest whole new concept areas to explore.

Knowledge Reveals Design Feasibility or Other "Llity" Assessments. A third category of ways that LLM agents justified $K \rightarrow$ C transitions was when knowledge revealed feasibility or other "ility"

Table 2 Qualitative codebook: major themes in LLM agents' rationale describing why $K \rightarrow C$ transitions occurred

Primary codes	Secondary codes	Definitions
Knowledge acts as a creative springboard	Knowledge space *reveals design gaps, recombinations, or analogies* that suggest new design directions.	Knowledge reveals design gaps or opportunities based on structured taxonomies of relevant existing designs, including via combination with seemingly unrelated or near/far analogous concepts.
	Knowledge space *reveals design trade-offs or contradictions* that suggest new design directions.	Knowledge reveals tensions like design performance trade-offs or fundamental design contradictions (e.g., simplicity versus universality, protectionversus usability, integration versus modularity) that expose fertile ground for design reframing or refinement.
	Knowledge space *reveals design feasibility or other "ility" assessments* that suggest new design directions.	Knowledge confirms or questions aspects of concept feasibility, usability, viability, sustainability, context suitability, etc., thus suggesting new design directions (e.g., moving from "can it work?" to "what new forms can it take?" or if it cannot work or is not suitable for the context, "what other approaches are possible?").
Concept lacks definition		The concept needs more definition as it is too broad or ambiguous to be interrogated against or further refined using the knowledge space.
C-K space metacognition		The LLM agent reflects on the state of its concept and knowledge spaces, reflects on where emergent ambiguity lies and how well the current spaces have been explored, and then moves accordingly between K and C, shifting between applying knowledge and generating new concepts.

assessments that pointed to new design directions. For example, consider the following rationale statement:

> Knowledge confirms that a disposable clip-on nail holder is technically feasible and economically viable, which shifts the design challenge from, *"can it exist?"* to *"what other possibilities does this enable?"*. In C-K theory, such validated knowledge does not close the process but instead opens new conceptual directions (e.g., reusable, eco-friendly, multi-functional variations). Therefore, the correct next step is $K \rightarrow$C, using feasibility insights as a springboard to expand the concept space with richer design alternatives.

In this example, knowledge helped to illustrate the feasibility and viability of a concept, which, in turn, opened new conceptional design directions. However, we found that infeasibility and other negative "ility" assessments also motivated transitions from $K \rightarrow$C. Another LLM agent justified a $K \rightarrow$C shift by stating that:

> Because the new knowledge reveals a fundamental limitation: magnetic holders only work with ferromagnetic nails. This is not a detail that can be solved by further testing ($K \rightarrow$K) but instead opens a new design requirement: compatibility with non-ferromagnetic nails. Therefore, the knowledge gained triggers the need to generate new concepts exploring alternative holding mechanisms beyond magnetism.

Thus, we found that both positive and negative "ility" assessments could propel LLM agents to shift from $K \rightarrow$C.

6.2.2 Concept Lacks Definition. In another major theme, we found that LLM agents made $K \rightarrow$C transitions when they could not effectively engage a given concept in their knowledge space due to it being too broad or ambiguous or the LLM's knowledge space being too limited. These instances were concentrated in the early time-steps of our simulation runs before many concepts were added to the starter knowledge base. This is illustrated by the following rationale statement:

> The current concept of a nail holder is still too broad and ambiguous to be validated against the existing knowledge base, which [currently] contains general design process knowledge rather than domain-specific insights. Since there are many unexplored solution directions (mechanical, material, ergonomic, etc.), the appropriate step is to expand the concept space ($K \rightarrow$C) to generate more concrete sub-concepts. This exploration will reduce ambiguity and later enable meaningful integration into the knowledge space.

Here, the LLM agent recognized that its knowledge space was too limited and the concept was too broad to allow for refinement without more conceptual definition back in the concept space.

6.2.3 C-K Space Metacognition. Beyond knowledge acting as a creative springboard, we also found that LLM agents justified moving from $K \rightarrow$C through a process of self-reflection or metacognition on the contents of their concept and knowledge spaces (perhaps akin to a human designer reflecting on their own concepts, knowledge, and mental models). These instances involved the LLM agent assessing where emergent ambiguity was present in their concept and knowledge spaces and how well the current knowledge base had been explored. Then they moved accordingly between K and C, shifting between applying knowledge and generating new concepts. To illustrate, consider the following rationale statement:

> The knowledge space is already rich with validated insights (K6, K14) on safety, adaptability, precision, durability, and usability, so further $K \rightarrow$K expansion would only yield incremental refinements. Instead, these validated elements now serve as creative building blocks that can be recombined to inspire novel, unexplored design directions. Therefore, a $K \rightarrow$C is the correct move to prevent premature closure and open new pathways for innovative nail holder concepts.

Here, the LLM agent reflected on the current state and richness of its knowledge space as a way to understand where to go next. In similar metacognition-oriented rationale statements, LLM agents stated: "In short, we now know enough to design, so the correct next step is to move from ($K \rightarrow$C)" and "the remaining ambiguity lies not in validating existing solutions ($K \rightarrow$K) but in exploring new possibilities." Taken together, LLM agents appeared to keep tabs on how much ambiguity was present in their concept and knowledge spaces and moved accordingly between the spaces.

6.3 Addressing Validity. Study 3, like other inductive qualitative theory building studies, aims for internal rather than external validity (i.e., transferability or generalizability). The results provide preliminary descriptions of why particular large language models (i.e., GPT-5) moved from $K \rightarrow C$ spaces when engaged in a design ideation task. Future research can explore the transferability of our findings to other LLMs, other interpretations or operationalizations of C-K theory, other conditions and design challenges, and, perhaps, to human designers. To establish internal validity, we adhered to recommended practices for analyzing qualitative data [59,60], including ensuring concept saturation with many replications of the unit of analysis ($n = 62$ in this case) and

having multiple coders who iterated a codebook to establish strong intercoder reliability.

6.4 Discussion. Study 3 was in part motivated by the findings in studies 1 and 2 that LLM agents' concept diversity decreased over time, especially when agents were stuck in continual $K \rightarrow K$ loops. To break from such loops, we identified $K \rightarrow$C transitions as particularly important to understand– not just whether and when $K \rightarrow C$ transitions occur, but how and why they occur. In study 3, we have uncovered preliminary categories of ways that an LLM agent used to rationalize moving from $K \rightarrow C$ when acting as a designer in a design ideation scenario. This contributes to the existing base of C-K theory by elaborating on the specific conditions that appear to trigger transitions from $K \rightarrow C$ in an algorithmic design context. Future work can similarly examine LLM agents' rationale for other transitions ($C \rightarrow K$, $K \rightarrow K$, and $C \rightarrow C$) and the degree to which these compare with the C-K transition rationale of human designers.

7 Discussion

7.1 What Our Large Language Model-Based C-K Simulation Does. Across all three studies, our direct implementation of C-K theory on top of an LLM converged to a specific pattern: long $C \rightarrow C$ loops, sparse and often unproductive $C \rightarrow K$ and $K \rightarrow C$ operations, few $K \rightarrow K$ operations, and a monotonic decline in concept space diversity. We also noticed that the agent never revisited earlier concepts or restructured the concept space. Thus, once a concept branch was created, it simply kept elaborating the branch in depth.

This behavior is close to how Agogué et al. describes a specific form of fixation that is driven by restrictive heuristics than to the generic generativity that is associated with splitting knowledge bases as described in Le Masson et al.'s Bauhuas analysis [35,40]. It also matches the prior work on GenAI that suggest contemporary LLMs are strong at $C \rightarrow K$ and $C \rightarrow C$ operations but are weak at $K \rightarrow C$ or $K \rightarrow K$ operations [39].

Methodologically, this means that a naive, prompt-level translation of the four C-K operators does not replicate human C-K reasoning as described in Hatchuel and Weil's work [20]. Instead, it primarily exposes the LLM's own generative tendencies under a C-K shaped protocol. Concepts are therefore only textual proxies for undecidable propositions, and the LLM-based agent is best understood as a constrained C-K like reasoner acting over a narrow knowledge space.

7.2 Implications for C-K as a Design Theory. The fact that the simulation collapses in this way does not invalidate C-K theory as a descriptive account of human design reasoning [20]. It does show, however, that C-K, as currently written, is not directly executable as an algorithm on top of a GenAI model. Our results point to what would have to be made explicit if one wants to turn C-K into an algorithmic simulation.

- *Knowledge splitting.* In our implementation, $K \rightarrow C$ was specified to operate as using knowledge to suggest a new concept. The LLM responded by reusing the existing knowledge to deepen the existing concepts. As noted by Le Masson et al., this does not yield generic generativity [35]. Instead, $K \rightarrow C$ must bring in structurally different pockets of knowledge, which will reveal contradictions and restructure the knowledge space so that new modes of exploration can appear [35]. Without explicit splitting policies, $K \rightarrow C$ will remain a recombination operator and diversity will keep shrinking, reproducing the restrictive-heuristic fixation patterns described in the prior work [40].
- *Reordering the concept tree.* Our simulation followed a purely reactive policy on the current item and never pruned, re-attached, or revisited concepts. The prior work on C-K theory assumes that designers can reorder the tree being developed in the concept space as knowledge is being split and reinterpreted [38]. A simulation that implements C-K theory would therefore need rules for when to abandon a branch, when to backtrack to earlier concepts in light of new knowledge, and how to re-position concepts under a newly structured knowledge, rather than committing to a single active path.
- *Knowledge reordering*. We allowed knowledge only to grow. Therefore, once a knowledge is added, none of the knowledge was renamed, redefined, or reorganized. The prior work has emphasized that design requires knowledge to be reordered to preserve meaning when new objects appear [38]. Thus, future C-K simulations would need to include explicit operations for updating the internal structure of the knowledge (e.g., the different knowledge categories and their relationships to each other) rather than just adding more textual propositions.

Thus, our contribution is to clarify the gap between C-K as a design theory and C-K in its computational simulation form. The four operators and their verbal descriptions are insufficient as a decision policy for an LLM agent. To obtain trajectories closer to what was described in follow-up studies on C-K theory (e.g., Refs. [35,38]), these missing mechanisms must be made operational and tied to concrete rules.

7.3 Large Language Model-Based Simulations as a Tool for Design Theory. We see LLM-based C-K simulations as stress tests and diagnostic tools. Because an LLM will follow the exact text of whatever protocols we encode (as seen from our case studies), without the tacit heuristics and meta-reasoning that human designers bring, it makes every underspecification in the theory visible, such as missing rules for when to leave $C \rightarrow C$, how to structure $K \rightarrow C$, and how to reorder the knowledge space [24]. At the same time, the simulation can create a generative profile—in our case strong $C \rightarrow C$ presence, weak $K \rightarrow K$ presence, and shallow $K \rightarrow C$ breadth—that makes fixation and lack of knowledge splitting a particularly noticeable issue [34,35,40]. In that sense, simulations, even when they are naively implemented, can be informative precisely because they can potentially fail and reveal where additional operational structures are necessary.

Overall, our case study points to an opportunity for future design theory work. Our implementation shows that LLMs, left to their own devices, will not spontaneously know how to do critical policies like knowledge splitting [35]. But the same experiments also reveal that they are already powerful, programmable operators for $C \rightarrow C$ and $C \rightarrow K$. Thus, rather than asking LLMs to be the designer and simulate designers, our results suggest treating them as configurable components inside a theory-guided architecture (e.g., some sort of algorithmic structure that handles knowledge splitting). Under such conditions, LLM-based simulations can prove to be especially valuable as stress-testing and diagnostic tools for design theories for the purpose of surfacing gaps, ambiguities, and fixation tendencies in a theory's operationalization before moving to more costly human or field studies.

8 Limitations

This study has many limitations. First, our results are quite sensitive to prompting choices and implementation details. Although we aimed to stay faithful to C-K theory in wording and operator logic, LLMs may require different phrasing or emphasis than a human reasoner. Future work could explore alternative prompting strategies and more explicit rule-based decision policies to examine whether the observed patterns (for example, dominant $C \rightarrow C$ transitions) persist under different formulations.

Second, the simulation is a narrow case study intended to illustrate potential, not to establish a definitive account of C-K theory. The task scope, as such, was limited to a single design brief and a small number of initial knowledge configurations. As such, our

results should be interpreted as exploratory. Broader studies spanning multiple design problems, larger knowledge bases, and different model architectures would be needed to test generality across contexts.

Third, our diversity evaluation relied on embedding-based measures that are standard in design ideation research but not necessarily comprehensive. Diversity, creativity, and novelty are multidimensional phenomena. As such, future work should examine alternative metrics, and where feasible, include human expert ratings for validation.

Finally, our computational mapping to C-K theory remains an oversimplification. Undecidability and "truth in K" were operationalized heuristically rather than through formal logic, and the implementation inevitably abstracts away aspects of human creativity. Our intent, however, is not necessarily to re-evaluate C-K theory itself but to demonstrate how LLM-driven simulations can function as stress tests for a design theory and to identify where design theories may benefit from LLM-driven simulations.

9 Conclusion

This article shows that LLMs can be used to make design theories executable and observable through simulation. By using C-K theory as a case study, we found that LLMs can be used in simulations to replicate a theory's basic logic, which, in turn, can potentially reveal gaps in the theory itself. Our findings suggest that specific decision rules in C-K theory could potentially be underspecified. Alternatively, this approach can reveal that existing design theories may not be adapted to LLMs, suggesting a need to develop new or adapted design theories for algorithmic design processes. Taken together, our article suggests that LLM-based simulations have the potential to serve as stress tests for design theories to expose such gaps and clarify how design reasoning might unfold when conducted computationally, making LLM-based simulations a potentially powerful tool for evaluating and refining design theories.

Acknowledgment

This work builds upon prior work published at the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2025 [61]. Niharika Sharma, an undergraduate at the University of California, Berkeley, and affiliate of the CoDesign Lab, aided in the development of preliminary prototypes of the C-K theory simulation. This work helped inform future development of the prototype, and the authors give ample thanks to her for her contribution and hard work. In addition, the authors wish to thank the members of the Berkeley Institute of Design for their valuable feedback. This work was supported through funding from the Keith L. Markolf and Cristina T. Ruland Fellowship.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The data and information that support the findings of this article are freely available[3].

Appendix A: Penalizer Prompts

```
You are a C-K Theory expert. In the input, I provided {number_concept} sets of consecutive C→C operations where each group of concepts are separated by a comma and contain a concept title and a concept description. In a C→C operation, the concepts are expanding by partitioning and exploring new attributes. Do these expanding concepts still hold significant ambiguity or unexplored potential that cannot yet be resolved or validated? Note, that this path is chosen when further ideation or exploration is necessary to refine the concept or when the concept introduces novel elements that challenge existing knowledge boundaries. Focus on determining whether there exists truly novel propositions that can transform or extend the knowledge space? Are the iterations becoming repetitive in its idea and content? These are questions you ask yourself whiledetermining whether to say Yes or No on whether or not we should continue doing C→C operation.
```

Appendix B: Human-Centered Design Knowledge Entries

```
  {
''id": ''K1",''type": ''knowledge",
''title": ''Identifying Customer Needs",
''desc": ''Understanding and documenting what customers truly
  require from a product. This involves interviews, observations,
  and translating customer statements into actionable needs."
''operation_rationale": ''Does not apply here because this is an
  initial knowledge."
},
{
''id": ''K2",
''type": ''knowledge",
''title": ''Concept Generation",
''desc": ''Developing a wide range of potential solutions that
  could satisfy the identified needs, encouraging both creative
  and structured ideation approaches."
''operation_rationale": ''Does not apply here because this is an
  initial knowledge."
},
{
''id": ''K3",
```

[3]See Note 2.

```
''type": ''knowledge",
''title": ''Concept Selection",
''desc": ''Systematically evaluating and narrowing down concepts
  using criteria such as customer needs, technical feasibility,
  and cost."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K4",
''type": ''knowledge",
''title": ''Prototyping and Testing",
''desc": ''Building representations of product concepts to test
  functionality, usability, and manufacturability. Can range from
  low-fidelity mockups to working prototypes."
''operation_rationale": ''Does not apply here because this is an
  initial knowledge."
},
{
''id": ''K5",
''type": ''knowledge",
''title": ''Design for Manufacture and Assembly (DFMA)",
''desc": ''Optimizing the product design for ease and cost-
  effectiveness of manufacturing and assembly, while maintaining
  performance."
''operation_rationale": ''Does not apply here because this is an
  initial knowledge."
}
```

Appendix C: System Knowledge Entries

```
  {
''id": ''K1",
''type": ''knowledge",
''title": ''Common User Errors in Hammering",
''desc": ''Users often miss the nail and strike their fingers,
especially at task initiation; other scenarios of impact risk (e
.g., rebound, sideways slip) could also be examined as part of
understanding injury modes."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K2",
''type": ''knowledge",
''title": ''Functional Requirements",
''desc": [
''The holder is typically expected to securely hold a nail in a
vertical or intended orientation until it is sufficiently
embedded to stand independently."
''The holder is typically expected to have the hammer contact the
nail head without interference."
''The holder is typically expected to release the nail easily once
partially driven."
],
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K3",
''type": ''knowledge",
''title": ''Variety of Nail Sizes and Shapes",
''desc": ''The holder should accommodate a representative range of
common nails (for instance, roughly 20 to 100mm long), though
designs targeting specialized nail sizes or variable holding
geometry may also be considered."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K4",
```

```
''type": ''knowledge",
''title": ''Material and Durability Considerations",
''desc": ''Materials are usually expected to resist hammer impact
and provide good grip; softer, replaceable, or energy-absorbing
materials could also be examined."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K5",
''type": ''knowledge",
''title": ''Human Factors and Ergonomics",
''desc": ''The holder must be operable with one hand, accommodate
left- and right-handed users, and require minimal fine motor
skill or grip strength."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K6",
''type": ''knowledge",
''title": ''Environmental and Operational Context",
''desc": ''The holder will be used in diverse settings indoors,
outdoors, or in variable conditions though extreme or
specialized contexts could motivate unique design adaptations."
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
},
{
''id": ''K7",
''type": ''knowledge",
''title": ''Verification and Validation Criteria",
''desc": [
''Under typical operation, the device should prevent direct contact
between the user's fingers and the hammer trajectory; however,
alternative safety mechanisms that achieve equivalent protection
could also be verified."
''Nail stability and alignment prior to impact are expected
performance indicators, but additional criteria such as ease of
nail release or accuracy after repeated use may also be
evaluated depending on concept variants."
''Assessment of task completion time, user comfort, and perceived
safety through usability testing provides baseline validation."
],
''operation_rationale": ''Does not apply here because this is an
initial knowledge."
}
```

Appendix D: Temporal Replication Summary

To check for temporal variability, we repeated the human-centered design knowledge priming simulation condition 1 week after the initial runs using the same codebase, prompts, and parameters. The repeated simulations produced qualitatively similar trends where there were declining diversity and alternating $C \rightarrow C$ and $K \rightarrow C$ transitions but differed in exact transition sequences. These differences likely result from inherent stochasticity in the model and potential client-side model updates in the hosted LLM environment. Figure 7 summarizes these additional runs. We treat this as a reproducibility observation rather than a new experiment.

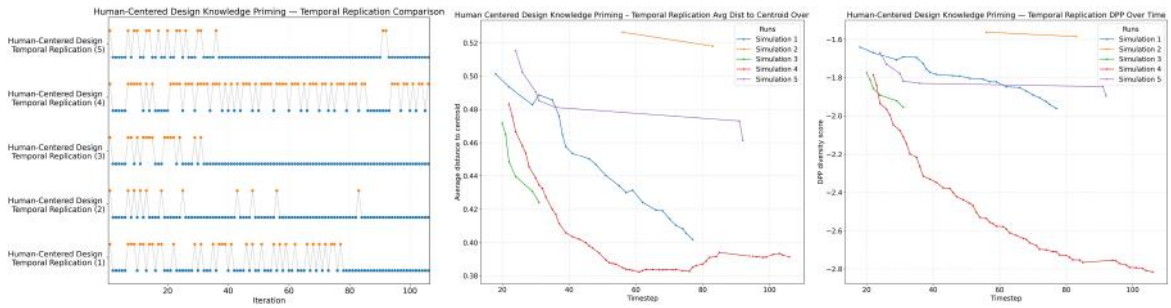

Fig. 7 Exploratory temporal replication results showing variation in C–K transition trajectories across repeated runs

References

[1] Hazelrigg, G. A., 1999, "On the Role and Use of Mathematical Models in Engineering Design," ASME J. Mech. Des., **121**(3), pp. 336–341.
[2] Weber, R. C., and Davis, R. L., 1982, "A Mathematical Model Used in Conjunction With a Finite Element Analysis to Aid in the Design and Analysis of Bourdon Tubes," ASME J. Mech. Des., **104**(3), pp. 540–543.
[3] Tangsali, K., Krishnamurthy, V. R., and Hasnain, Z., 2021, "Generalizability of Convolutional Encoder–Decoder Networks for Aerodynamic Flow-Field Prediction Across Geometric and Physical-Fluidic Variations," ASME J. Mech. Des., **143**(5), p. 051704.
[4] Song, B., Yuan, C., Permenter, F., Arechiga, N., and Ahmed, F., 2024, "Data-Driven Car Drag Prediction With Depth and Normal Renderings," ASME J. Mech. Des., **146**(5), p. 051714.
[5] Marchesse, Y., Changenet, C., Ville, F., and Velex, P., 2011, "Investigations on CFD Simulations for Predicting Windage Power Losses in Spur Gears," ASME J. Mech. Des., **133**(2), p. 024501.
[6] Wacker, J. G., 1998, "A Definition of Theory: Research Guidelines for Different Theory-Building Research Methods in Operations Management," J. Oper. Manage., **16**(4), pp. 361–385.
[7] Wacker, J. G., 2008, "A Conceptual Understanding of Requirements for Theory-Building Research: Guidelines for Scientific Theory Building," J. Supply Chain Manage., **44**(3), pp. 5–15.
[8] Briggs, R. O., 2006, "On Theory-Driven Design and Deployment of Collaboration Systems," Int. J. Hum-Comput. Stud., **64**(7), pp. 573–582.
[9] Newell, A., and Simon, H. A., 1972, *Human Problem Solving*, Prentice-Hall, Englewood Cliffs, N.J..
[10] Simon, H. A., 1969, *The Sciences of the Artificial, 3rd Edition*, 1st ed., Vol. 1, The MIT Press, Cambridge, MA.
[11] Winograd, T., and Flores, F., 1987, *Understanding Computers and Cognition: A New Foundation for Design*, Addison-Wesley, Reading, MA.
[12] Suchman, L. A., 1987, *Plans and Situated Actions: The Problem of Human-Machine Communication*, Cambridge University Press, Cambridge, UK.
[13] Fernandes, J. V., Henriques, E., Silva, A., and Pimentel, C., 2017, "Modelling the Dynamics of Complex Early Design Processes: An Agent-Based Approach," Des. Sci., **3**, p. e19.
[14] Meluso, J., and Austin-Breneman, J., 2018, "Gaming the System: An Agent-Based Model of Estimation Strategies and Their Effects on System Performance," ASME J. Mech. Des., **140**(12), p. 121101.
[15] Lapp, S., Jablokow, K., and McComb, C., 2019, "KABOOM: An Agent-Based Model for Simulating Cognitive Style in Team Problem Solving," Des. Sci., **5**, p. e13.
[16] Repenning, N. P., 2001, "Understanding Fire Fighting in New Product Development," J. Prod. Innovation Manage., **18**(5), pp. 285–300.
[17] March, J. G., 1991, "Exploration and Exploitation in Organizational Learning," Organ. Sci., **2**(1), pp. 71–87.
[18] Wilson, R. C., and Collins, A. G. E., 2019, "Ten Simple Rules for the Computational Modeling of Behavioral Data," eLife, **8**, p. e49547.
[19] Davis, J. P., Eisenhardt, K. M., and Bingham, C. B., 2007, "Developing Theory Through Simulation Methods," Acad. Manage. Rev., **32**(2), pp. 480–499.
[20] Hatchuel, A., and Weil, B., 2009, "C-K Design Theory: An Advanced Formulation," Res. Eng. Des., **19**(4), pp. 181–192.
[21] Gero, J. S., and Kannengiesser, U., 2004, "The Situated Function–Behaviour–Structure Framework," Des. Stud., **25**(4), pp. 373–391.
[22] Chen, L., Zuo, H., Cai, Z., Yin, Y., Zhang, Y., Sun, L., Childs, P., and Wang, B., 2024, "Toward Controllable Generative Design: A Conceptual Design Generation Approach Leveraging the Function–Behavior–Structure Ontology and Large Language Models," ASME J. Mech. Des., **146**(12), p. 121401.
[23] Jiang, S., and Luo, J., 2024, "AutoTRIZ: Artificial Ideation With TRIZ and Large Language Models," ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Washington, DC, Aug. 25–28.
[24] Hatchuel, A., and Weil, B., 2003, "A New Approach of Innovative Design: An Introduction to C-K Theory," DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, A. Folkeson, K. Gralen, M. Norell, U. Sellgren, eds., Stockholm, pp. 109–110, Paper No. DS31_1794FPC.
[25] Hatchuel, A., Masson, P. L., and Weil, B., 2004, "C-K Theory in Practice: Lessons From Industrial Applications," International Design Conference – DESIGN 2004, Dubrovnik, Croatia, May 18–21, pp. 245–258.
[26] Kazakci, A. O., and Tsoukiàs, A., 2005, "Extending the C–K Design Theory: A Theoretical Background for Personal Design Assistants," J. Eng. Des., **16**(4), pp. 399–411.
[27] OpenAI, 2024, "GPT-4 Technical Report," 2303.08774, https://arxiv.org/abs/2303.08774
[28] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. U., and Polosukhin, I., 2017, *Attention Is All You Need*, Vol. . 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., Curran Associates, Inc., Red Hook, NY, pp. 5998–6008.
[29] OpenAI, 2025, "OpenAI GPT-4.5 System Card," PublicationSafety, This is an OpenAI publication.
[30] Zhu, Q., and Luo, J., 2024, "Toward Artificial Empathy for Human-Centered Design," ASME J. Mech. Des., **146**(6), p. 061401.
[31] Ma, K., Grandi, D., McComb, C., and Goucher-Lambert, K., 2025, "Do Large Language Models Produce Diverse Design Concepts? A Comparative Study With Human-Crowdsourced Solutions," ASME J. Comput. Inf. Sci. Eng., **25**(2), p. 024501.
[32] Baudoux, G., and Goucher-Lambert, K., 2024, *Automating Analogical Reasoning: A Wizard of Oz Study on the Benefits and Pitfalls of a Sketch-Based AI Image Generator for Design*, J. S. Gero, ed., Springer, Cham, pp. 21–37.
[33] Chen, L., Xia, D., Jiang, Z., Tan, X., Sun, L., and Zhang, L., 2025, "A Conceptual Design Method Based on Concept—Knowledge Theory and Large Language Models," ASME J. Comput. Inf. Sci. Eng., **25**(2), p. 021001.
[34] Bordas, A., Le Masson, P., Thomas, M., and Weil, B., 2024, "What Is Generative in Generative Artificial Intelligence? A Design-Based Perspective," Res. Eng. Des., **35**(4), pp. 427–443. Published 09 October 2024.
[35] Le Masson, P., Hatchuel, A., and Weil, B., 2016, "Design Theory at Bauhaus: Teaching "Splitting" Knowledge," Res. Eng. Des., **27**(2), pp. 91–115.
[36] Wiener, N., 1948, *Cybernetics: Or, Control and Communication in the Animal and the Machine*, MIT University Press, Cambridge, MA.
[37] Chattoe, E., 1998, "Just How (Un)realistic Are Evolutionary Algorithms as Representations of Social Processes?" J. Artif. Soc. Soc. Simul., **1**(3).
[38] Hatchuel, A., Weil, B., and Le Masson, P., 2013, "Towards an Ontology of Design: Lessons From C–K Design Theory and Forcing," Res. Eng. Des., **24**(2), pp. 147–163.
[39] Bordas, A., Le Masson, P., and Weil, B., 2025, "Switching Perspectives: Enhancing Generative Artificial Intelligence's Creativity, by Humans, With Design," Creativity Innovation Manage., **34**(4), pp. 1013–1033.
[40] Agogué, M., Poirel, N., Pineau, A., Houdé, O., and Cassotti, M., 2014, "The Impact of Age and Training on Creativity: A Design-Theory Approach to Study Fixation Effects," Thinking Skills Creativity, **11**(1), pp. 33–41.
[41] Regenwetter, L., Srivastava, A., Gutfreund, D., and Ahmed, F., 2023, "Beyond Statistical Similarity: Rethinking Metrics for Deep Generative Models in Engineering Design," Comput.-Aided Des., **165**, p. 103609.
[42] Doshi, A. R., and Hauser, O. P., 2024, "Generative AI Enhances Individual Creativity But Reduces the Collective Diversity of Novel Content," Sci. Adv., **10**(28), p. eadn5290.
[43] Jia, M., Jiang, S., Hu, J., and Qi, J., 2023, "Toward Understanding Sources and Influences of Design Fixation: A Focus on Example Stimuli and Background of Novice Designers," ASME J. Mech. Des., **145**(5), p. 051402.
[44] Leahy, K., Daly, S. R., McKilligan, S., and Seifert, C. M., 2020, "Design Fixation From Initial Examples: Provided Versus Self-Generated Ideas," ASME J. Mech. Des., **142**(10), p. 101402.
[45] Crilly, N., and Cardoso, C., 2017, "Where Next for Research on Fixation, Inspiration and Creativity in Design?" Des. Stud., **50**, pp. 1–38.
[46] Vasconcelos, L. A., Cardoso, C. C., Sääksjärvi, M., Chen, C. -C., and Crilly, N., 2017, "Inspiration and Fixation: The Influences of Example Designs and System Properties in Idea Generation," ASME J. Mech. Des., **139**(3), p. 031101.
[47] Goucher-Lambert, K., and Cagan, J., 2019, "Crowdsourcing Inspiration: Using Crowd Generated Inspirational Stimuli to Support Designer Ideation," Des. Stud., **61**, pp. 1–29.
[48] Fu, K., Chan, J., Cagan, J., Kotovsky, K., Schunn, C., and Wood, K., 2013, "The Meaning of "Near" and "Far": The Impact of Structuring Design Databases and the Effect of Distance of Analogy on Design Output," ASME J. Mech. Des., **135**(2), p. 021007.
[49] OpenAI, 2025, "Introducing GPT-5," https://openai.com/index/introducing-gpt-5/, Accessed August 2025.
[50] Norman, D. A., 2002, *The Design of Everyday Things*, Basic Books, New York.
[51] Nielsen, C. B., Larsen, P. G., Fitzgerald, J., Woodcock, J., and Peleska, J., 2015, "Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions," ACM Comput. Surv., **48**(2), pp. 1–41.
[52] Honnibal, M., Montani, I., Landeghem, S. V., and Boyd, A., 2020, "spaCy: Industrial-Strength Natural Language Processing in Python, If You Use spaCy, Please Cite It as Below."
[53] Miles, M. B., Huberman, A. M., and Saldaña, J., 2018, *Qualitative Data Analysis: A Methods Sourcebook*, 4th ed., SAGE Publications, Inc., Thousand Oaks, CA, Arizona State University, USA.
[54] Hsieh, H.-F., and Shannon, S. E., 2005, "Three Approaches to Qualitative Content Analysis," Qual. Health Res., **15**(9), pp. 1277–1288.
[55] Saldaña, J., 2009, *The Coding Manual for Qualitative Researchers*, Sage Publications Ltd, Thousand Oaks, CA.
[56] Charmaz, K., 2006, *Constructing Grounded Theory*, Sage Publications, Thousand Oaks, CA.
[57] Braun, V., and Clarke, V., 2006, "Using Thematic Analysis in Psychology," Qual. Res. Psychol., **3**(2), pp. 77–101.
[58] Cohen, J., 1960, "A Coefficient of Agreement for Nominal Scales," Educ. Psychol. Meas., **20**(1), pp. 37–46.
[59] Weston, C., Gandell, T., Beauchamp, J., McAlpine, L., Wiseman, C., and Beauchamp, C., 2001, "Analyzing Interview Data: The Development and Evolution of a Coding System," Qual. Soc., **24**(3), pp. 381–400.
[60] Thomas, D. R., 2006, "A General Inductive Approach for Analyzing Qualitative Evaluation Data," Am. J. Eval., **27**(2), pp. 237–246.
[61] Ma, K., Goucher-Lambert, K., and Brubaker, E. R., 2025, "Simulating Design Theory Using LLM Agents: A Case Study of C–K Theory," Proceedings of ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC), Anaheim, CA, Aug. 17–20.