Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Introducing Grounding into the Agentic Design Loop: An Exploratory Review of an Unclaimed Problem

Zhongkai Wang
Tongji University
Yan Liu
yanliu.sse@tongji.edu.cn

Tongji University

Research Article

Keywords: Agentic design, Design grounding, Large language models, Exploratory review, Cross-domain analysis

Posted Date: November 12th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11444119/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Introducing Grounding into the Agentic Design Loop: An Exploratory Review of an Unclaimed Problem

Zhongkai Wang[1] and Yan Liu[1*]

[1]School of Computer Science and Technology, Tongji University, Shanghai, China.

*Corresponding author(s). E-mail(s): yanliu.sse@tongji.edu.cn;
Contributing authors: 2251640@tongji.edu.cn;

Abstract

Large language model (LLM) based agents are moving from executing well-specified tasks to Agentic Design: shaping what and how to build upstream of fabrication, integration, and deployment. However, an internally coherent but externally infeasible design propagates its compounding error downstream. The failure is easy to overlook, yet consequential and hard throughout: a design-generating loop rarely includes a check against the world, and the error surfaces only after the artifact is built. Design Grounding names the correspondence between such a design and the world it must satisfy. The mechanisms that establish it exist in adjacent fields but have been introduced into the design loop only in isolated, domain-specific forms, which leaves the same difficulty scattered across fields under different names. Studying both sides through their own corpora, the review is the first to consolidate them into one unclaimed problem. What makes the problem hard is structural: a design's referent is a system that has not been built, and the model that produces the design cannot supply it without external intervention. The review is exploratory in aim: it characterizes the problem and screens Agentic Design and Grounding in all four combinations. Reading titles and abstracts with an LLM flags roughly one in fifteen of nearly three thousand papers as closing the Design Grounding loop; only twenty survive full-text adjudication and manual review. Those twenty sort into four structural difficulties, and each becomes an open problem whose parts already exist in an adjacent field but remain unassembled.

Keywords: Agentic design, Design grounding, Large language models, Exploratory review, Cross-domain analysis

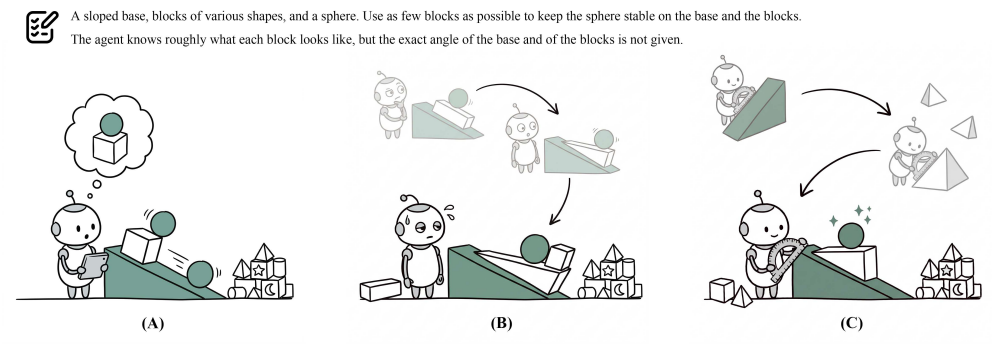


Fig. 1 Three states of one design task. The same task appears in all three, differing in what the agent can call on before the design is fixed. The instruction given to the agent, shown at the top, is the same in all three panels. Friction between the blocks and the base is assumed to be sufficient, so the sphere is the only element that moves, and the panels are schematic rather than a physics benchmark. **(a) The model alone.** The design comes from the model's own competence and nothing returns from the world: a single rectangular block is placed on the slope, the sphere rolls off, and the failure appears only after the build (Section 4.1). **(b) Built, then told otherwise.** The agent learns the verdict only after the build: the rectangular block fails, a wedge whose open slant still lets the sphere roll off fails, and a second block wedged above the sphere finally holds it. The signal is now independent of the agent, but it arrives after the design has been fixed, so the result takes several rounds, uses two blocks where one would do, and leaves the top unlevel (Section 4.3). This loop closes by trial and error, and a failed attempt can simply be rebuilt. However, the design settings the review covers offer no such guarantee: an attempt there may be expensive to repeat, may be irreversible, or may already be in the field. **(c) Grounding, then built.** The agent measures the slope of the base and of the candidate blocks before building and picks the single piece whose angle matches the base, which turns the slope into a level step; the sphere rests on it at the first attempt. The measurement is the grounding, and it enters the design rather than the artifact. This panel marks the direction the survey points to (Section 5), not a result it reports. The illustration was created with an AI image generation tool. The task icon was made by UIcons from Flaticon.

1 Introduction

An expanding landscape of **Agentic Design** is taking shape: LLM-based agents (Wang et al. 2024a; Xi et al. 2025) are moving beyond executing well-specified tasks into making *upstream* design decisions, generating specifications for engineered systems (International Organization for Standardization 2015; INCOSE 2023). These decisions already range over hardware architectures, software system topologies, and other targets. The agent does more than implement, and it has gradually begun to decide *what and how to build*.

This movement is the latest wave in the development trajectory of an agentic technology that began with language modeling itself. Early LLMs demonstrated that explicit reasoning steps could elicit reliable answers (Chain-of-Thought) (Wei et al. 2022), and pairing that reasoning with external tools turned generation into action (ReAct, Toolformer) (Yao et al. 2023; Schick et al. 2023). Agents then gained the ability to reflect on their own failures (Reflexion) (Shinn et al. 2023), to collaborate in multi-agent settings (CAMEL, Generative Agents, MetaGPT) (Li et al. 2023; Park

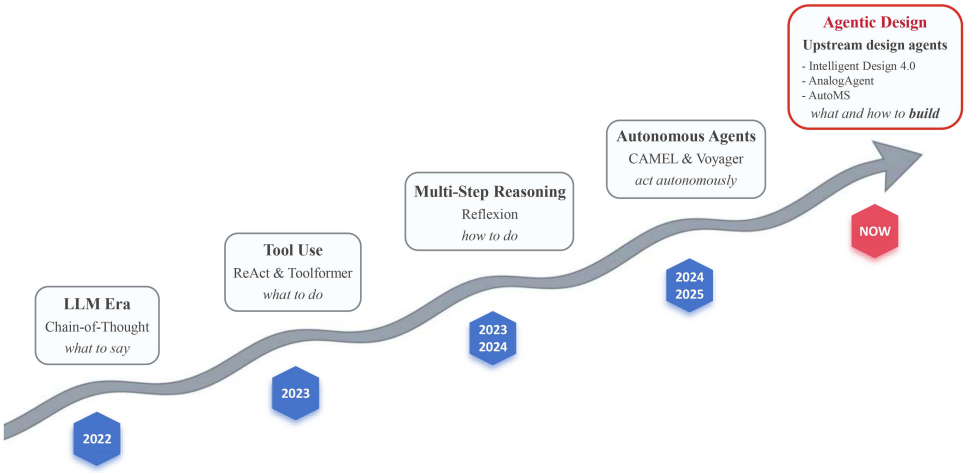


Fig. 2 From language modeling to Agentic Design. The development of agentic systems is drawn as a rising wave. The stages span Chain-of-Thought reasoning (Wei et al. 2022), tool use with ReAct (Yao et al. 2023) and Toolformer (Schick et al. 2023), multi-step reasoning with Reflexion (Shinn et al. 2023), autonomous multi-agent operation with CAMEL (Li et al. 2023) and Voyager (Wang et al. 2023), and Agentic Design, where upstream design agents decide what and how to build, exemplified by Intelligent Design 4.0 (Jiang et al. 2025), AnalogAgent (Bao et al. 2026), and AutoMS (Zhao et al. 2026). The "NOW" marker indicates that Agentic Design is the current frontier of this trajectory. The wave band in the background was generated with an AI image generation tool.

et al. 2023; Hong et al. 2024), and to operate autonomously in open-ended environments (Voyager) (Wang et al. 2023). With each wave, the object the agent manipulates moved further from text and closer to the world: first tokens, then tool calls, then long-horizon behaviors, and now system specifications that must hold in a physical or operational environment (Intelligent Design 4.0, AnalogAgent, AutoMS) (Jiang et al. 2025; Bao et al. 2026; Zhao et al. 2026). Figure 2 summarizes this trajectory.

Work along this trajectory is already accumulating, and the direction of travel is clear. What has not followed is validation of the resulting designs: the upstream decisions go unexamined. The difficulties that come with the move are reported where they arise, among them evaluation gaps in software engineering (Hou et al. 2024; Fan et al. 2023), hallucination in language tasks (Ji et al. 2023; Huang et al. 2025), a simulation-to-reality gap in robotics (Zhao et al. 2020), and model fidelity in scientific computing (Do and Zhang 2023). Each community treats its difficulty as local to its field, but *no survey has yet placed them side by side*. Reading the design-oriented agent papers and the difficulties they report as one body of evidence, this review identifies a common structural pattern (Section 2). The dominant mode of evaluation is benchmark-based: the agent generates a design, and the design is assessed through internal metrics, proxy scores, or human judgment. What is missing, however, is *real-world validation*. The difficulties share a common root: the agent's internal judgment of design quality is not calibrated against the external reality that the design must ultimately satisfy. An internally coherent design may be externally infeasible, and

the agent, lacking exposure to that external reality, has no mechanism to detect the divergence.

This observation points to a deeper question:

When an agent's design drifts away from the world it must satisfy, what brings it back?

The question is long-standing, widely explored, and scattered across fields under a single name: **Grounding**.

Grounding, in the broad sense inherited from the Symbol Grounding Problem (Harnad 1990), refers to the process of establishing and maintaining correspondence between a system's internal representations and the external world through observations, interactions, or externally anchored feedback (Cohen et al. 2024; Kenthapadi et al. 2024). The traditions that study it provide mechanisms for this process, each anchored to a different target: perception (Cadena et al. 2016), language and fact (Lewis et al. 2020), action (Ahn et al. 2022), plan, and state, as the cross-domain analysis in Section 2 shows. However, the grounded object is seldom the agent's Design: in the two corpora analyzed in Section 2, the mechanisms that could ground a design have not been introduced into the loop that produces it. Unsurprisingly, the loops that do close use tools built for a single domain.

Taken together, Design and Grounding have developed in parallel, and the two occupy opposite ends of the engineering lifecycle, which runs from the design that fixes what will be built to the operation that tests it. Design lives at the upstream, symbolic end: it is a statement in a language, easy to generate, easy to revise, and easy for an LLM-based agent to produce, which is precisely why it drifts from reality. Grounding lives at the downstream, physical end: the conditions a design must satisfy resist symbolization, and this resistance is what makes grounding hard to delegate to a machine. That distance makes a division of labor look natural, and each field has kept to its own end. Nevertheless, the two carry what the other lacks: a design without grounding is easy to produce but hard to trust, while the grounding traditions have anchored objects that exist and can be pointed at, none of which is a design that has not yet been built. In the loop, though, the two ends are where they come closest. Introducing grounding into the design loop is what would make it stronger. The loop would gain a criterion it cannot generate on its own, since the referent of a design does not yet exist and the judgment that produced the design is the judgment being tested.

That criterion has to come from a world the agent does not control. When an agent generates a design, that design carries assumptions about the world it will operate in. The **External World**, encompassing the physical, computational, or operational environment that the design must ultimately satisfy, exists independently of the agent's reasoning (Lee 2008). The design does not exist yet, but the world it must satisfy does. That world is encountered, observed, or discovered rather than constructed to evaluate the agent, so it cannot be asked in advance whether a design will work. The gap between an agent's internal design judgment and the external realizability of that design is the **Internal-to-External Bias**.

Stated as a problem, with d a design, W the External World, and $\mathrm{Bias}(d, W)$ the divergence between the agent's internal judgment of d and the verdict that W would give: find an evaluation signal S and a revision operator R such that

$$d_{t+1} = R\big(d_t,\, S(d_t)\big), \qquad \mathrm{Bias}(d_T, W) < \mathrm{Bias}(d_0, W), \tag{1}$$

where S does not depend on the agent's own judgment, $S(d)$ arrives before d is committed, and R can change the structure of d. The part of the loop where this reduction happens is its internal-to-external slice.

Searching both sides turns up near misses rather than a junction (Section 3): one side builds loops that end inside the agent, and the other develops returns from the world that rarely enter a design process. Grounding a design is still rare, and the connection between the two sides has remained unnamed. Existing reviews cover each side on its own: those of agentic systems map their architectures, capabilities, and applications (Wang et al. 2024a; Xi et al. 2025; Abou Ali et al. 2025), with design-facing reviews staying inside single domains such as hardware (Pan et al. 2025), algorithm design (Liu et al. 2026a), and scientific discovery (Gridach et al. 2025). Reviews of grounding, in turn, cover its traditions in robotics, language, and fact (Cohen et al. 2024; Kenthapadi et al. 2024). *None of them treats that connection as a problem in its own right, and making that explicit is the purpose of this survey.*

Design Grounding, therefore, is the name this survey gives that problem. In the sense used here, Design Grounding occurs when a signal anchored in the External World is used to evaluate an agent-generated design and to revise, select, reject, or terminate it. Figure 1 stages that bias on a task small enough to follow: with the model alone, nothing returns from the world; a verdict that arrives only after the design is fixed closes the loop from the outside, and every correction costs another build; and a measurement taken before the build brings the signal back while revision is still cheap.

Scope. This survey explores that slice of Design Grounding, and it reads that slice from the digital side: the agent, the design, and the loop that evaluates them are the terms it names, while the physical world enters only as the referent the design must satisfy. How designs are generated inside the agent, and how the External World changes on its own, enter the picture only as they affect it. The perspective from the physical side, in which an artifact is fabricated, instrumented, and operated, stays outside the account. That boundary is a limitation of the survey, and it also marks the part that can be worked on first while the problem is this new. The design loop, the cycle in which an agent proposes, evaluates, and revises a design, already runs in Agentic Design, and it can appear complete while its evaluation stays inside the agent.

The two names this survey uses for it, the design loop and the Design Grounding loop, mark two states of that one cycle: it becomes a Design Grounding loop once the signal it is evaluated against is independent of the agent and returns while the design can still change, and Section 3.1.3 grades what that signal is made of. Design Grounding, as defined here, is introduced in this survey, and it names a gap rather than an established field. The phrase differs from grounded design in design research, where grounding means deriving a design from empirical evidence rather than checking it against the world it must satisfy.

Figure 3 draws the contrast. A human designer walks the site, measures what is actually there, and revises the plan while revision is cheap; an agent producing the

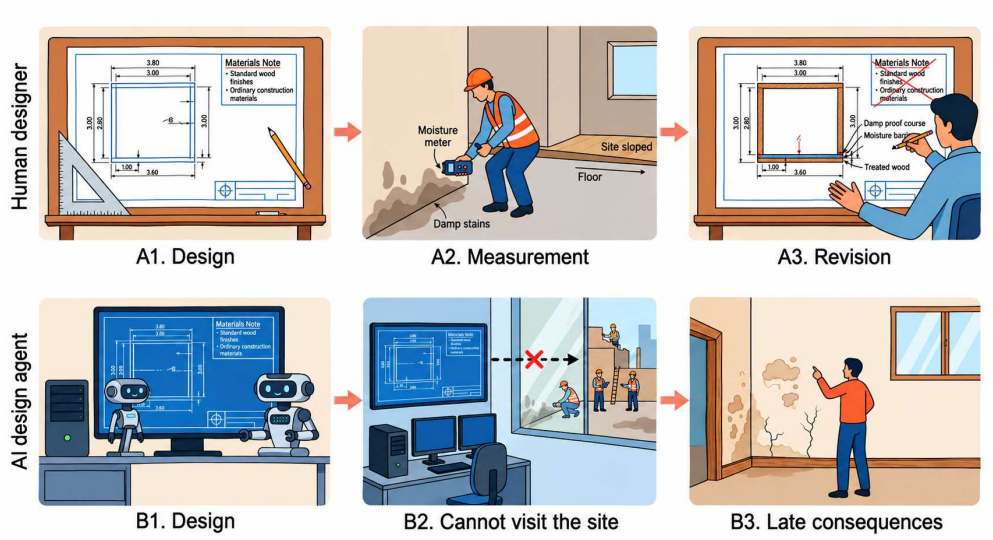


Fig. 3 From the drawing board to the site, and back. In the top row, a human designer walks the site, measures what is actually there, and revises the plan before construction. In the bottom row, an agent designs the same renovation without such a visit, so nothing returns from the site, and the mismatch surfaces only after the house is built, in the form of damp damage. The illustration was created with an AI image generation tool.

same plan has no such visit in its loop, and the mismatch surfaces only after the house is built. The plan assumes dry walls; the site says otherwise. Human contact with a site is sensory and continuous, while an agent touches the environment only through the channels it has been given, and what is not instrumented does not enter its loop. Design Grounding, in this sense, is everywhere in design practice but missing from the agent's usual loop.

That absence has concrete consequences: design decisions sit at the upstream end of the engineering lifecycle, so their effects propagate through components, interfaces, implementation, testing, and deployment. An internally coherent but externally infeasible design can still be implemented faithfully, but a design error then becomes a system-level failure whose cost is paid downstream. *The later an error is discovered, the more expensive the rework* (Tan et al. 2017) (Section 4.3). As agents take on design decisions in more settings, these stakes grow. Design Grounding matters because it determines whether early decisions remain connected to the environment they must ultimately satisfy. Establishing that connection is worth the effort, and the means to do it already exist: a design error is the most leveraged kind of error, adjacent fields supply the mechanisms, and a few loops have already moved toward it.

Keeping that connection is not straightforward. The difficulty of Design Grounding is **structural**. Designs are generated upstream of implementation, so the External World can only be observed through signals that are

- **partial**: the conditions that determine realizability, such as physical constraints, resource budgets, deployment environments, and user behavior, are not fully specified in the design model and cannot be enumerated in advance (Krakovna et al. 2020; Pan et al. 2022);
- **delayed**: feedback, when it arrives, may come from a simulator, a prototype, or a deployed system, each with different fidelity and coverage;
- **indirect**: turning that feedback into a design revision is a control problem that few existing agent architectures address, since the revision must preserve the integrity of the original specification without overfitting to the available signal.

These difficulties are not specific to any one domain, and they recur wherever agents make design decisions without direct access to the environment those decisions will encounter. At this stage, the field needs a clearer picture of the difficulty, and drawing that picture is what this survey attempts in Section 4.1.

Our working hypothesis is that design-level grounding is scarce and scattered: few agent pipelines close a feedback loop that starts from the External World and returns to a design, and techniques that could close it exist in adjacent fields under other names. Placing Agentic Design and Grounding side by side is what turns that distance into evidence of what the design loop is missing: the open problems that follow are collected, not solved, in Section 5, and the four-quadrant screening of Section 3 separates design from grounding before it measures their overlap, so the intersection is counted rather than assumed.

The survey proceeds in two movements.

- **Exposing the gap**: Section 2 uses cross-domain evidence to show where Agentic Design has expanded, why its difficulties recur, and how far Grounding research stands from the Design level.
- **Mapping the territory**: Section 3 describes a structured search across arXiv, DBLP, and grey literature, together with the four-quadrant screening framework and the audits that test its reliability, and the resulting corpus grounds the challenge analysis of Section 4, the open questions of Section 5, and the validity discussion of Section 6.

The examination carries a definitional difficulty of its own: what counts as design and what counts as grounding are questions the literature does not answer uniformly, and the survey treats them as decisions to be examined rather than terms to be assumed. This survey is exploratory: the questions came first, the search was built to let the evidence answer them, and the conclusions follow from what the corpus did and did not contain.

Our contributions are exploratory, and they are meant as a map of the problem:

1. To reveal the difficulty each community keeps to itself as one cross-domain gap, we characterize Design Grounding as an unclaimed problem, connecting discussions across several fields that have developed in relative isolation. We also show why a design-stage error is the most leveraged kind of error and the most expensive to correct.

2. Through a four-quadrant systematic search, we quantify the intersection between Agentic Design and Grounding, separating work that forms a Design Grounding loop from work that lacks one or more of its components. With such a count, the scarcity of grounded design becomes a measured quantity.
3. From a cross-domain analysis of design-oriented agent papers and grounding research, we extract the recurring difficulties that Agentic Design faces without Grounding, show how seldom the grounding side reaches the Design level, and examine the near-miss patterns that reveal where the loop almost closes. The difficulties are reported by the papers themselves, and their recurrence across domains is what turns the gap from an impression into a documented pattern.
4. We identify transferable grounding mechanisms from robotics, NLP, embodied AI, and related areas, and examine the challenges of adapting them to the Design level. Design Grounding has no established toolbox of its own, so the mechanisms of adjacent fields are the nearest starting points to try at the Design level.
5. We derive open questions and research directions for grounding-aware Agentic Design, spanning External World interfaces, feedback architectures, structural design revision, evaluation benchmarks, and cross-domain transfer. Because the gap has only just been made explicit, the field's immediate need is a well-posed list of open problems, and these directions mark where the evidence allows work to begin.

2 Agentic Design and Grounding: A Gap in Both Directions

This section examines the gap between Agentic Design and Grounding with cross-domain evidence from two analyzed corpora. It tests three claims in turn, each corresponding to an observation of the Introduction: agents have moved into upstream decisions, their difficulties recur across domains, and the grounding traditions have not followed them there. The first two claims concern the design side of the loop, and the third concerns the grounding side: together they establish that the return the loop needs exists elsewhere and has not been brought into it. The claims are addressed through three questions:

1. **The upstream move.** Are agentic systems expanding into upstream design decisions across domains?
2. **The common condition.** Are the recurring difficulties of Agentic Design consistent with an absence of real-world feedback, and is that feedback available in the design setting?
3. **The missing counterpart.** Does Grounding research reach the Design level?

The evidence in this section comes from the screening pipeline documented in Section 3, and the order of the two sections follows the order in which the analysis developed. The gap came first: reading design-oriented agent papers alongside grounding papers showed where the return from the world was missing, and that reading set the questions the pipeline had to answer. The pipeline was therefore iterative. It was built around those questions and revised as the papers it returned exposed borderline cases, because the boundary between a design that is grounded and one that merely

reports a metric is not fixed in advance; inclusion rules, quadrant labels, and corpus definitions were adjusted accordingly. In brief, it is a PRISMA-style search over arXiv, DBLP, and grey literature, with LLM-assisted screening, four-quadrant classification, full-text adjudication, and a human audit. Section 3 documents the final pipeline and its counts; this section reports what the two corpora show about the gap.

2.1 Data and Method Overview

To answer these questions, we analyze the design-oriented agent papers and the grounding papers selected from that pipeline:

- The **Design-only (DO)** corpus contains papers whose primary artifact is a design: 184 papers, drawn from the 215 design-only papers identified in the screening pipeline. The quadrant is defined by the absence of feedback into the design rather than by the absence of grounding, and 33 of the 184 have a grounding component that does not return to the design. The remaining 31 papers could not be retrieved: 30 have no arXiv version and are reachable only through the publisher, and one failed to download. This limitation is discussed in Section 6.
- The **Grounding-only (GO)** corpus contains papers whose research core is Grounding: 192 papers, sampled from the 1,229 grounding-only papers with equal allocation across the six most frequent grounding targets, 25 papers each, and the rest drawn from the remaining targets. Full text was retrieved for all 192.

  Operational definitions. The suffix "only" indicates the quadrant a paper occupies in the four-quadrant screening described in Section 3: a DO paper exhibits Agentic Design whose feedback does not update the design, whether or not a grounding component is present, and a GO paper exhibits Grounding without an agent-generated Design. These are definitions introduced in this survey, not established terms in the literature.

Each corpus is read for what its side can show: the DO corpus for the difficulties that arise when a design loop runs without a return from the world, and the GO corpus for the targets grounding research works on and the mechanisms it uses to reach them.

For each paper, an LLM assistant (qwen3.7-max) extracts the design artifact, the application domain, the reported difficulties, and the evaluation strategy. The analysis feeds the assistant the title, the abstract, and the sections in which authors discuss limitations, difficulties, and future work, or the closing part of the text when no such section is identified, and the assistant returns structured labels in JSON. The extraction is restricted to statements that appear in the text, so the reported difficulties come from the papers themselves rather than from the reviewer. The membership of both corpora, with the extracted fields for each paper, is provided in Online Resource 1.

2.2 Design-only Research Is Expanding Across Domains

The DO corpus shows that agents are being applied to design tasks in at least eight domains, and that these applications concentrate in the last two years. The coverage is bounded, however. Apart from robotics and control, the domains are subfields

Table 1 Distribution of the 184 Design-only papers by application domain and year. Agentic Design has expanded from 5 papers in 2023 to 93 in 2026, with the largest share in agent architecture, other engineering domains, and scientific computing. The 2026 column covers January to July.

Application domain	2023	2024	2025	2026	Total
Agent architecture	2	5	11	30	48
Other engineering	1	4	14	15	34
Scientific computing	0	3	10	17	30
Hardware EDA	1	3	8	9	21
Software engineering	1	0	9	11	21
AI model design	0	2	7	1	10
Other	0	1	5	4	10
Robotics and control	0	1	3	3	7
Network and telecom	0	0	0	3	3
Total	5	19	67	93	184

of digital technology, and wider design fields such as mechanical, civil, or chemical engineering barely appear. What digital domains offer that the others do not is machine-readable specifications and fast, cheap feedback, the conditions under which design agents currently thrive. The digital tilt is partly the survey's own: the search ran largely over computer science venues, a limitation examined in Section 6. Table 1 lists the distribution by domain and year.

Three observations follow from this table.

1. *The growth is recent and rapid.* The corpus grows from 5 papers in 2023 to 93 in 2026, and 160 of the 184 papers appear in 2025 and 2026.
2. *The coverage is broad.* No single domain dominates: the largest, agent architecture, accounts for 48 of 184 papers, and the remaining papers are spread across engineering, science, and computing domains.
3. *The expansion crosses artifacts that differ in form but share a structural position.* Hardware EDA papers produce accelerator architectures; scientific computing papers produce synthesis routes and experiment workflows; software engineering papers produce module structures and coding-agent harnesses; and agent architecture papers produce coordination protocols and team compositions. These artifacts fix decisions before implementation begins, which is the position we call the **Design level**.

Answer to Q1. Agentic systems are moving into upstream design decisions. The breadth and the recency of this expansion make the connection between designs and external reality a first-order question.

2.3 Real-World Feedback Is Missing, and Hard to Obtain Even When Sought

If design-only papers are expanding, the next question is *how they are evaluated*. Two pieces of evidence address this question: the evaluation strategy reported in each paper, and the difficulties each paper reports. Table 2 summarizes the evaluation strategies,

and Table 3 the reported difficulties. The four strategies come from grouping the free-text descriptions of how each paper evaluates its design, and they are coarser than the difficulty categories that follow.

Table 2 Evaluation strategies in the 184 Design-only papers. Most papers evaluate their designs through benchmarks, simulations, or internal metrics; only 33 of 184 report any form of real-world validation.

Strategy	Papers	Share
Benchmark or simulation only	103	56.0%
Internal metrics or proxies	48	26.1%
Benchmark and real-world	20	10.9%
Real-world validation only	13	7.1%

Table 3 Reported difficulties in the 184 Design-only papers. The most frequent categories, scalability, reasoning limitation, generalization, and domain knowledge, describe the gap between internal evaluation and external validity. Each paper's difficulty was classified by an LLM from the paper's own one-sentence limitation statement; the full list appears in Online Resource 2.

Difficulty	Papers	Share	Description
Scalability	28	15.2%	Works on small instances, fails as the problem or system grows
Reasoning limitation	23	12.5%	Shallow or over-anchored reasoning without system-level foresight
Generalization	17	9.2%	Fails in settings different from the evaluation setting
Domain knowledge	16	8.7%	Lacks domain-specific or physical constraints
Semantic correctness	13	7.1%	Output lacks verifiable correctness, consistency, or feasibility
No difficulty extracted	12	6.5%	No explicit difficulty statement
Cost or efficiency	11	6.0%	Cost, latency, or resource overhead limits the design process
Simulation-to-reality gap	11	6.0%	Internal evaluation diverges from external behavior
Lack of evaluation	10	5.4%	No evaluation is reported
Data scarcity	8	4.3%	Insufficient domain data for the design task
Other	8	4.3%	Does not fit the categories above
Expert dependence	7	3.8%	Human designers supply constraints and validation
Hallucination	7	3.8%	Coherent but externally infeasible outputs
Long horizon	5	2.7%	Consequences unfold over long time scales
Feedback quality	5	2.7%	Feedback is partial, delayed, or noisy
Structural change	3	1.6%	Revisions arrive late and propagate through components

The evaluation picture is uneven. The majority of the corpus, 151 of 184 papers, assesses designs through benchmarks, simulations, or internal metrics. Only 33 papers report any form of real-world validation. The pattern is consistent with what one would expect when a design is generated and evaluated entirely within the agent's own loop, without the design being realized in the environment it targets. The strategy recorded here is what a paper measures, which is a different question from the grounding grade of Section 3.1.3: a simulation evaluation supports grounding at G1 or G2 when the

signal it produces returns to the design, and the Design-only quadrant is defined by the absence of that return rather than by the use of simulation. Meanwhile, the reported difficulties reinforce the same conclusion from the other direction. For each paper, the difficulty is a one-sentence statement that the LLM assistant extracted from the paper's own limitations discussion. The sixteen categories are set out in advance and applied by the assistant, which returns a category and a one-line reason for each paper, with a residual Other label for those that fit none of them; the category definitions and the classification prompt are reproduced in Appendix A, and the statements with their categories and reasons appear in Online Resource 2.

The extraction labeled more papers as having no difficulty statement than the corpus supports, in part because the papers vary in where they state them. The section slicer identifies Limitations, Discussion, and Conclusion sections by their headings, and it falls back to the closing text of the paper when no heading matches, which happens for a paper that places its limitations under another heading, such as Threats to Validity, or in an appendix after the reference list. No fixed rule covers every layout the corpus contains, and for those papers the extractor received references or appendix material instead of the discussion. The 23 papers in that bucket were re-examined against their full texts, and 11 of them state a limitation the pass had missed, which leaves 12 papers (6.5%). The recovered statements were classified with the same categories; where a single sentence does not carry the category, the assignment follows the surrounding passage. The twelve that remain report results and future directions without naming a limitation of their own method, and one of them discusses only the shortcomings of earlier work. The other categories of Table 3 were not re-checked, since a paper usually states more than one limitation and a second reading can select a different one, which carries a different category; Section 6.6 returns to what that means for the table.

Reading these statements directly, the five most frequent name one condition from different angles, the Internal-to-External Bias of Section 1: *scalability* (a design that works on small instances fails as the problem or system grows), *reasoning limitation* (the reasoning is shallow or over-anchored, without system-level foresight), *generalization* (the design fails in settings different from the evaluation setting), *domain knowledge* (the agent lacks domain-specific or physical constraints), and *semantic correctness* (the output lacks verifiable correctness, consistency, or feasibility). Each names a way in which that bias appears: the agent's internal judgment of a design is not calibrated against the external reality the design must satisfy. The statements are the authors' own, so they record what the field notices, and what they share in cost is that the error arrives later than the decision and propagates to everything built from it. These statements recur across domains in the DO corpus.

Two categories name the grounding problem directly: *simulation-to-reality gap* and *feedback quality* are reported by papers that do not close the loop, which is consistent with the quadrant, since the authors recognize the condition even where their systems leave it unresolved.

Two further categories are notable for how rarely they appear. *Hallucination* appears in only seven papers, which is unexpected for the failure mode most often attributed to the models themselves. The seven statements describe outputs that are

coherent but externally infeasible, the same condition that semantic correctness and the simulation-to-reality gap describe. The failure is therefore spread across three rows, and none of them names it as fabrication. *Long horizon* is the rarest category, at five papers, and the one where engineering work has the most room to expand: decisions whose consequences unfold over years are exactly where no current loop keeps the consequence attached to the decision, and the cost floats until it lands in operation.

Together they describe a practice in which designs are generated and assessed largely without contact with the environment they must satisfy. The papers that report no difficulty explicitly, or that report difficulties outside these categories, are discussed in Section 6.

Answer to Q2. Real-world feedback is scarce in Agentic Design and hard to obtain. A design practice that generates and assesses designs without contact with the environment leaves no channel through which external validity can enter.

2.4 Grounding Research Seldom Targets Design

The third question turns to the other side of the gap: *where Grounding research has been active, and what it grounds*. The GO corpus provides the landscape. Table 4 summarizes the application domains of the 192 sampled papers, and Table 5 the grounding targets they address.

Table 4 Application domains of the Grounding-only corpus (192 sampled papers) by year. Grounding research spans agent tool execution, embodied agents, and domain science, with a longer history than Agentic Design. The 2026 column covers January to July.

Domain	2020	2021	2022	2023	2024	2025	2026	Total
Agent tool execution	0	0	1	1	7	17	42	68
Embodied agents	1	3	1	3	2	16	10	36
Domain science	0	0	0	0	3	6	20	29
Game and simulation	0	0	0	1	2	8	9	20
NLP factuality	0	0	0	0	0	4	9	13
Dialogue and instruction	0	0	1	0	1	6	3	11
Visual perception	0	0	0	0	0	2	8	10
Other	0	1	0	0	1	0	2	4
State estimation	0	0	0	0	0	1	0	1
Total	1	4	3	5	16	60	103	192

The GO corpus shows where Grounding research has been active. The dominant application domains are agent tool execution, embodied agents, and domain science. Within the sample, the dominant grounding targets are action, text or fact, and perception, the traditional objects of Grounding across robotics, NLP, and vision (Ahn et al. 2022; Cadena et al. 2016; Lewis et al. 2020). The sample was allocated evenly, so these shares describe the sample rather than the field. The target recorded during screening came from the assistant in the same pass as the quadrant judgment and serves only to define those strata, whereas the target reported here comes from a set of seven categories written and defined by the authors, one sentence each, and applied by

Table 5 Grounding targets in the Grounding-only corpus (192 sampled papers), with a re-check of a stratified sample of 34. Action, text or fact, and perception dominate; the seven design cases are boundary conditions close to task artifacts rather than upstream system specifications. The allocation is even across targets rather than proportional to the pool, so these shares describe the sample and not the field. Targets are the re-judged labels, not the allocation strata. The re-check column gives the number of sampled papers that a blind full-text reading by a different model placed in the same target, with the design and other strata sampled in full and the rest at about one in eight. It measures how far the recorded labels reproduce under a second reading, and it is not a frequency over the corpus, since the sample was drawn by target.

Target	Papers	Re-check
Action	96 (50.0%)	6 of 12
Text or fact	40 (20.8%)	5 of 5
Perception	17 (8.9%)	3 of 3
Task plan	16 (8.3%)	2 of 3
State or world model	15 (7.8%)	3 of 3
Design	7 (3.6%)	2 of 7
Other	1 (0.5%)	1 of 1
Total	192	22 of 34

the second judgment to every sampled paper. Where the two disagree, the movement favors action: the 25 implementation-code strata were read as grounding on execution in 21 cases, and component-level labels follow the same direction once the fuller text is read, with perception 10 of 25, task plan 13 of 25, and state or world model 9 of 25. The strata record what a paper names in its abstract, while the second judgment records what the system is calibrated against, and for a perception module, a planner, or a code generator that calibration usually runs through execution. A re-check tested the two labels the GO corpus reports. A stratified sample of 34 papers was read in full by deepseek-v4-pro, a different model family from the one that produced the labels, blind to the recorded labels and with the definitions above, and it placed 22 of them in the same grounding target and 30 in the same application domain, so the domain labels are the stable part of the extraction and the target labels the approximate one. The disagreements follow the target boundaries rather than scattering: four of the twelve action papers were read as perception, and five of the seven design papers were placed elsewhere, three of them in other. Design is therefore the stratum where the two readings differ most, and the re-check moves papers out of it rather than into it, which leaves the count of seven as the upper end of its range.

Aside from the residual Other category, design is the rarest target the sample names, with seven of the 192 papers, and even those seven are boundary cases rather than core design research: they include agents that ground their outputs in specification documents or in the results of prior analysis steps, where the grounded object is closer to a task artifact than to an upstream system specification.

Why does Grounding research skip design? The composition of its targets offers a hint. Action, the dominant target, is what can be probed cheaply: an action is tried, fails fast, and is corrected in the same loop, so grounding naturally flourishes where the loop is short. Design sits at the opposite pole: probing a design means building the system it specifies, so the loop is long wherever the check has to reach the world

the design must satisfy. A design can be checked cheaply against itself, by internal consistency or a small trial, and what such checks cannot catch is the error whose cost arrives with the system. An error caught inside a trial costs one iteration, while an error that survives design is paid for over the life of the system. Under a regime of fast, low-cost trial and error, one could argue that upstream design is unnecessary, and that iteration alone could replace it. The DO corpus shows where the practice already stands: agents make upstream design decisions at scale, so the design stage is occupied now, and whether iteration could absorb it later is a separate question rather than a reason to leave the stage unexamined. The hypothesis, whether feedback-rich iteration could eventually absorb the design stage, is kept open and revisited in Section 5.

The temporal pattern of the GO corpus differs from that of the DO corpus. Grounding work appears as early as 2020, and its growth predates the Agentic Design wave by several years. The two lines of research are therefore both growing, but they have grown in parallel rather than toward each other. Each line lacks what the other provides. The DO corpus grows without real-world feedback, while the GO corpus develops feedback mechanisms without targeting design.

Therefore, the GO corpus reveals what Grounding research is missing: an absence of design as a target, alongside an abundance of effort on other targets.

> **Answer to Q3.** Grounding research has a longer history than Agentic Design, yet it has not reached the Design level. Its feedback mechanisms are directed at action, perception, and fact, and design remains outside their scope.

2.5 The Portrait of the Gap

Taken together, the three tables draw a consistent portrait. Agentic Design is expanding (Table 1), and its designs are assessed mostly without real-world feedback. Real-world feedback is also hard to obtain, as the difficulty evidence in Table 3 shows. Meanwhile, Grounding research develops the mechanisms that provide such feedback, through observations, interactions, and externally anchored signals, yet it does not cover the Design level (Table 5). The gap between the two lines of research is therefore real, growing, and unaddressed: the Design side expands year over year, while the Grounding side stays with its traditional objects.

This portrait motivates the systematic examination in Section 3. If the gap is real and growing, the questions that follow concern the work that does try to close it: *how much of it exists, what forms it takes, and where the loop almost closes but misses.*

3 Mapping the Design-Grounding Landscape

The preceding section showed that Agentic Design is expanding while Grounding research does not reach the Design level. This section turns the qualitative picture into a quantified landscape and documents the pipeline that produced the corpora of Section 2. Through a systematic search and a four-quadrant screening pipeline, we measure how much work directly connects Design to an External World through feedback, and we examine the boundary cases where that connection almost closes. Figure 4 summarizes the complete pipeline and the numbers at each stage.

3.1 Search and Screening Method

LLM-assisted screening has become an established practice in systematic review automation, with evaluations reporting substantial acceleration of title and abstract screening and a remaining need for human oversight (Khraisha et al. 2024; Wang et al. 2024b; Huotala et al. 2024). The pipeline below follows this practice and adopts the PRISMA reporting structure, which recent methodological work has extended to GenAI-assisted workflows (Shailendra et al. 2026).

3.1.1 Sources and Query Design

We searched three sources: arXiv, restricted to the cs.AI, cs.SE, cs.CL, cs.RO, and cs.LG categories; DBLP, used as a unified index of major computer science publishers, conferences, and journals; and grey literature, including research institution blogs, agent framework documentation, and coding-agent design documents. This mix follows from three considerations.

- **arXiv** is where this line of work appears first: method papers and system reports circulate as preprints ahead of, or instead of, archival publication.
- **Grey literature** records design decisions that papers only summarize, if they mention them at all, such as how a coding agent's feedback loop is wired. The search covered three source types: agent framework documentation (19 records), coding-agent design documents (17), and research-institute blogs (14). Each record was audited manually: 64 raw records were collected, 14 were dropped, and 50 were retained, of which 46 had their URL and title verified and 4 were kept although the domain could not be resolved.
- **DBLP** anchors the search to the published, peer-reviewed record, so claims from the faster layers stay checkable against it.

Semantic Scholar was used solely to complete missing abstracts.

The queries were organized into four families, 128 queries in total, covering the period from January 2020 to July 2026 with no per-query limit, and the searches were last executed on 19 July 2026. The 2026 counts reported in this survey therefore cover seven months of the year, and they are lower bounds rather than full-year totals. Table 6 summarizes the families and their roles, and the complete list of 128 queries appears in Appendix B.

The raw records were cleaned and deduplicated in five steps:

1. versionless arXiv identifiers;
2. normalized DOIs;
3. reverse extraction of arXiv identifiers from DOI-based arXiv records;
4. exact standardized titles;
5. fuzzy title matches resolved by manual confirmation.

Of the 4,658 raw records, arXiv contributed 3,642, DBLP 966, and the grey literature 50. After 258 records were removed as duplicates, 4,400 candidates remained. After abstract completion through arXiv and Semantic Scholar, 4,231 candidates

Table 6 The four query families and their roles. Each family combines agent terms with a different set of concept terms; the Design-centric family does not require Grounding terminology.

Family	Queries	Role	Example concept terms
Grounding × Agent	48	Work involving both Grounding and agents	grounding, grounded, affordance, sim-to-real
Synonym × Agent	56	Adjacent formulations	hallucination, calibration, verification, factuality
Grounding-only	4	Transferable Grounding foundations	grounding, symbol grounding, reference
Design-centric	20	Agentic Design without Grounding terms	design, inverse design, design space exploration, co-design

had abstracts and entered abstract-level screening, while the 169 without abstracts proceeded directly to full-text screening.

3.1.2 LLM Four-Quadrant Screening

Working definitions. The two screening stages apply the working definitions introduced in Section 1, which define Agentic Design, Grounding, and Design Grounding. The quadrant stage judges whether each dimension is present, and the full-text stage judges whether the complete loop closes. The prompts used in the two stages are reproduced in Appendix A.

Role of the models. The pipeline uses two main models with complementary roles.

- qwen3.7-max handles the lightweight tasks: the scope and quadrant screening of the large candidate pool, the domain, difficulty, and evaluation extraction for the Section 2 corpora, and the difficulty classification behind Table 3, for which a fast, accurate abstract-level judgment suffices.
- deepseek-v4-pro, which reasons more effectively over full texts and applies a looser interpretation of the inclusion criteria, performs the full-text adjudication, the rescoring of medium- and low-confidence records, and the full-text audit of the non-core quadrants. The looser interpretation is deliberate: it casts a wide net so that borderline cases reach the human reviewer.

All screening and extraction stages were run at temperature zero, and the screening was executed between 22 July and 2 August 2026. The model identifiers are those the provider exposed during that window and are used here as given; they carry no separate version tag. Providers repoint or retire aliases without notice, so the durable record of what each model contributed is the prompts in Appendix A and the per-paper outputs in Online Resource 3, which allow each judgment to be re-checked from the text without re-running the models. All final decisions, however, are made by the lead researcher (the first author) through manual review of the full texts.

A further question is whether the conclusions depend on the model choices behind the pipeline. The design keeps the risk limited. The decisive step, the quadrant assignment, is computed by a deterministic script from the model's field judgments rather

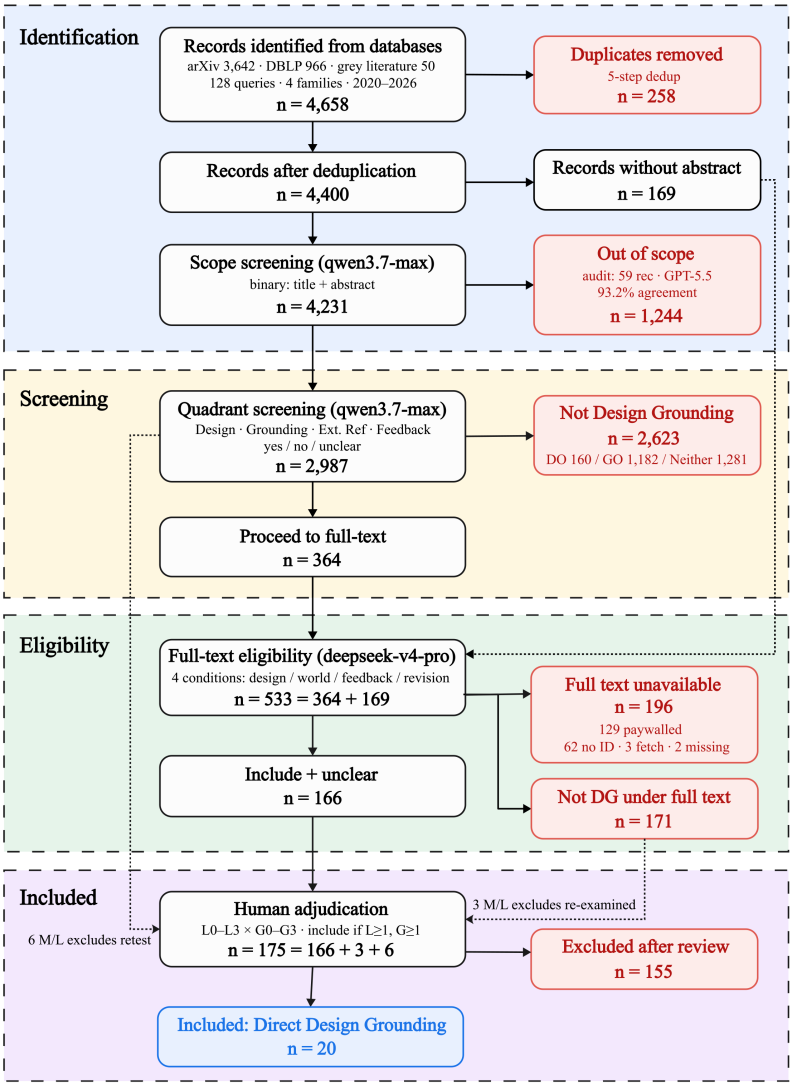


Fig. 4 The screening pipeline. A PRISMA-inspired flow from 4,658 identified records to the 20 papers of the final corpus. The four stage labels on the left mark Identification, Screening, Eligibility, and Included. Central boxes give the operation, the model used, and the number of records at each step; boxes on the right give exclusion reasons and counts. The dashed line from the quadrant stage marks the cross-model re-scoring of 91 medium- or low-confidence records by deepseek-v4-pro: six were promoted to the manual review but all were excluded afterwards, which confirms that the low-confidence judgments of the first model leave the final corpus unchanged. The 364 records routed to full text are the Design Grounding quadrant in full (199) together with 165 records from the other three that were unclear in at least one field or carried a logic warning; the remaining 2,623 in-scope records are Design-only at 160, Grounding-only at 1,182, and Neither at 1,281. The manual review covered 175 papers, composed of 166 from the full-text assessment, 3 medium-confidence excludes re-examined, and 6 promoted by the retest.

than asked of the model. Cross-model checks were built in at the points where errors would matter most, and where the two models disagreed, the disagreements did not survive the manual review. Every decision that determined the final corpus passed the manual review of the full text, so the survey's conclusions rest on human judgment at the boundary rather than on any single model. A different model would therefore be unlikely to change the conclusions of this survey, though it could plausibly shift individual counts.

Scope screening. The 4,231 records with abstracts first passed a scope filter that determined whether a paper belongs to LLM-based agent research. qwen3.7-max produced a binary include or exclude decision from the title and abstract: 2,987 included, 1,244 excluded. The excluded side was audited, because the scope filter is the gate that admits every record the later stages can consider. A stratified sample of 59 records was drawn from the excluded pool, part of it proportional to the strata and part of it a targeted oversample of the high-risk ones, and re-judged from the same titles and abstracts by gpt-5.5, a third model used only for this check. Agreement with the original decisions was 93.2%, and four of the 59 were re-judged as within scope. None of the four concerns an agent that generates a design, so none reaches the quadrant this survey assesses and the final corpus is unaffected.

The second stage, quadrant screening, classified the 2,987 included records along two dimensions: Design and Grounding. For each paper, qwen3.7-max output three-valued judgments (yes, no, unclear) for four main fields:

- Whether the paper involves Agentic Design
- Whether Grounding is present
- Whether an External World reference exists
- Whether feedback updates the design

The Grounding, External World reference, and feedback fields describe one relation with two legs. The **outbound leg** is the agentic process reaching toward the External World for evidence about the design, and the **return leg** is that evidence coming back and revising the design. *The return leg presupposes the outbound one.* Nothing returns that was never sent, so the absence of the outbound leg entails the absence of the return.

A broad Grounding judgment proved too loose: it counted a calibration setup as Grounding even when the calibrated object was an action, a skill, or a plan, with no Design involved. Design Grounding requires feedback that targets the agent-generated Design (Section 1), which is narrower than the literature's sense of any anchoring of internal representations to the world. The feedback field now asks only whether an External World reference directly updates the Design. Because the return leg carries the outbound leg with it, that field alone separates Design Grounding from Design-only, and the Grounding field stays to define the Grounding-only quadrant, whose papers ground something other than a design. The provisional quadrant was computed from these fields by a *script*, not by the model: Design Grounding required an Agentic Design with feedback updating it, Design-only an Agentic Design without

such feedback, Grounding-only Grounding without Design, and Neither neither condition. Unclear marked cases where the title and abstract left the question open, and a missing mention was never taken as evidence of absence.

Of the 2,987 records, the four-quadrant distribution was: Design Grounding 199 (6.7%), Design-only 215 (7.2%), Grounding-only 1,229 (41.1%), and Neither 1,344 (45.0%). Design was judged yes in 414 papers, Grounding in 1,451, External Reference in 1,721, and Feedback updating Design in 200. The quadrant count and the field counts do not coincide, because the rule above pairs Design with feedback rather than with the broader Grounding field: 199 of the 414 Design papers fall in the Design Grounding quadrant, and the one remaining record with feedback updating the design was judged not to involve Design. The Design-only quadrant follows from the same rule, so it gathers papers by the absence of the return leg rather than by the absence of grounding: of its 215 records, 150 were judged to have no grounding, 36 to have grounding that does not update the design, and 29 to be unclear on that field. Confidence was high for 2,799 records, medium for 182, and low for 6, and 73 records carried contradictory logic warnings.

Medium- and low-confidence records. The quadrant screen marked 188 of the 2,987 records as medium or low confidence. For the 97 that had already entered full-text screening, the full-text result took precedence. For the remaining 91, deepseek-v4-pro re-scored them with the same quadrant prompt as a cross-model consistency check: would a second model, judging the same abstracts, reach the same quadrant and confidence? The check promoted six papers to Design Grounding, all of which then entered the manual review. None of the six survived it. Every one was excluded at the Design level, so the low-confidence judgments of the first model do not change the final corpus.

Audits of the non-core quadrants. A stratified sample of 80 records was drawn from the Design-only, Grounding-only, and Neither quadrants (20, 30, and 30) and re-classified on the full text with deepseek-v4-pro. Of the 80 records, 49 had a retrievable full text and form the valid set. Within the valid set, 42 agreed (85.7%), with agreement of 78.6% for Design-only, 88.2% for Grounding-only, and 88.9% for Neither. Agreement is lowest in the Design-only stratum, which is also the stratum closest to the Design Grounding boundary: the distinction turns on whether feedback updates the design, a relation that an abstract often leaves implicit. Within the 49 valid records, the full-text audit re-classified three as Design Grounding that the abstract-level screen had placed outside the quadrant, an abstract-level miss rate of 6.1%. The audit fed no records back into the pipeline: one of the three independently reached the manual review through the main flow and was excluded there, and the other two never reached it. The miss rate is therefore an upper bound on the abstract screen's error rather than a measured loss.

3.1.3 Full-Text Adjudication and Human Review

Records that were Design Grounding (199), judged unclear in at least one field (144), or flagged by a logic warning (73) proceeded to full-text eligibility, together with the 169 records without abstracts. Because the three groups overlap, the resulting pool contained 533 papers: 364 from quadrant screening and 169 without abstracts. A

logic warning marks a record whose field judgments contradict one another or the quadrant they imply, such as grounding without an external reference; the quadrant was recomputed from the fields, and the record proceeded to full text in either case. A full-text adjudication with deepseek-v4-pro determined whether each paper satisfied four main conditions:

- Whether an agent generates a Design
- Whether an External World reference exists
- Whether feedback from that world returns to the agent
- Whether the agent uses the feedback to modify, select, reject, or terminate the Design

These four conditions differ from the quadrant fields in two respects: they are judged on the full text, where the quadrant fields were judged on the abstract, and the last condition requires the agent to actually use the feedback, which the abstract-level feedback field only approximates. The adjudication outcomes are summarized in Table 7. Excludes made at medium confidence in this stage were not discarded outright; they were re-examined in the manual review. Of the 196 papers without retrievable full text, only 14 came from the Design Grounding quadrant, so the risk of missing true Design Grounding studies through unavailability is limited. Within that quadrant, 185 of the 199 records had a retrievable full text and 132 were eligible at the full-text stage, and 17 of the 20 included papers come from it. The other three come from the 144 records whose abstract-level grounding or feedback judgment was unclear, which the routing rule sends to full text by design rather than as a confirmed exclusion.

Table 7 Outcomes of the full-text adjudication on the 533-paper pool. The 196 failures could not be assessed because no full text was retrievable.

Outcome	Papers
Include	155
Exclude	171
Unclear	11
Failure (no retrievable full text)	196
paywalled	129
no DOI or arXiv ID	62
fetch error	3
missing arXiv PDF	2
Total assessed	533

All final decisions were made by the lead researcher through manual review of the full texts. The pool for this review contained 175 papers: 155 include, 11 unclear, 3 excluded at medium confidence in the full-text stage and therefore re-examined, and 6 promoted from the medium- and low-confidence retest. The review applied a two-level grading scheme, set out in Table 8.

A learned predictor is constructed by the design process rather than encountered in it, and a human check is supplied by a person. Neither is the External World of

Table 8 The two grading scales. Design records how much of the artifact's structure is settled before the agent acts, and Grounding records what the loop closes against. The left panel gives the four Design levels and the right panel the four Grounding grades.

Level	What the agent settles before it acts
L0	the structure is fixed and only values remain to be chosen, so the agent optimizes or tunes an existing artifact
L1	a catalogue or action space fixes the components and the agent composes them
L2	the artifact's type and its conditions are fixed, and the agent determines the structure within them
L3	no part of the structure is fixed, and the agent determines it from scratch

Grade	What the loop closes against
G0	no valid grounding, such as constructed benchmarks, learned models, or human-in-the-loop verification
G1	a natural-law model in idealized or generic form, without a calibrated account of the materials or processes at issue
G2	prediction of a modeling engine that represents the system's materials or processes as they stand, such as a process design kit, a synthesis flow for a real technology node, or a solver run with a technology-grade device model
G3	real physical measurement or real data entering the loop

Section 1, which exists independently of the agent, so the grade asks whether the signal is independent of the agent rather than how accurate it is; a well-calibrated surrogate therefore falls to G0 on that test. The four levels are ordinal annotations rather than validated categories. The gradations between them, L1 to L3 and G1 to G2, record how far a paper reaches along each scale; they were assigned by a single reader and are used as descriptions rather than as measured quantities. Two boundaries carry the analysis: whether the agent designs at all, L0 against L1 and above, and whether the loop closes against a model of the system or against a measurement, G2 and below against G3. Autonomous closure was a *hard requirement*, so grounding through human-in-the-loop feedback was not counted, and a paper was included when Design was at least L1 and Grounding at least G1.

The manual review confirmed 20 of the 175-paper pool: 18 of the 155 full-text inclusions, one of the 11 unclear records, and one of the three medium-confidence excludes. The gap between the full-text inclusions and the final corpus reflects the deliberate width of the full-text stage: deepseek-v4-pro cast a wide net, and the manual review then enforced the L/G criteria strictly. The top of the Design scale was re-examined seven weeks after the first pass, against the criteria stated above, together with the papers whose first-pass confidence was low or medium. Three labels were corrected. Two remain in the corpus at a lower level: the concrete barrier of Wang et al. (2026c) moves to L2, since its section profile is fixed and only the geometry and the reinforcement remain to be determined, and the ASIC agent of Allam et al. (2025) moves to L1, since it integrates a module into a pre-built system-on-chip template. The third, Bhandwaldar et al. (2026), decides values inside a design it is given rather than determining that design's structure, so it falls below the inclusion threshold and leaves the corpus. The final corpus contains 20 included papers and 155 excluded papers.

3.1.4 Lightweight Citation Check

Instead of full citation snowballing, we performed a lightweight citation check. Full snowballing was impractical for this corpus: 14 of 20 papers are from 2026 and none predates 2024. Backward snowballing would therefore mostly surface foundational tools such as simulators, PDKs, and LLM frameworks rather than Design Grounding

Table 9 Cross-tabulation of Design level and Grounding strength over the 175 human-reviewed papers. All 20 included papers satisfy Design $\geq$ L1 and Grounding $\geq$ G1; none of the 155 excluded papers does.

	G0	G1	G2	G3
L0	95	0	29	8
L1	8	1	2	1
L2	5	0	3	0
L3	10	0	13	0

studies, and forward snowballing is inapplicable given the recency of the included papers.

As a lightweight check, we extracted the reference lists of the 20 included papers and screened them against the pool of 4,400 candidates with an LLM assistant. The match rate was low: most references point to foundational tools and traditional methods, which fall outside the 4,400-candidate pool by design. The lead researcher screened the flagged citations against the inclusion criteria: none met them, and the closest candidates were traditional non-LLM methods, simulation tools, or early LLM hardware-design work without closed-loop feedback. The Design Grounding quadrant itself proceeded to full text in full, so the residual risk of missing studies lies in the other three quadrants, which the stratified audit covers. Combined with the broad query strategy and the audit agreement of 85.7%, we treat the residual risk of missing studies that ground a design directly as bounded rather than eliminated.

3.2 The Four-Quadrant Landscape

The screening pipeline places every candidate in one of four quadrants defined by the presence of Design and Grounding, as shown in Figure 5. The landscape that results is uneven in both scale and timing.

Neither dominates the pool at 45.0%: most LLM agent research is concerned with neither design nor grounding, which puts the intersection at the margin of a field with other priorities. Grounding-only, at 41.1%, is nearly six times the size of Design-only (7.2%): grounding has a long history and a large literature, while design-oriented agent research is younger and smaller. Design Grounding, at 6.7%, is the smallest of the four quadrants: it sits at the crossing of these two unequal lines.

The timing of the four quadrants differs sharply. Grounding-only work appears as early as 2020 and Design-only in 2023, echoing the timelines reported in Section 2. Design Grounding appears only in 2024, as Figure 6 shows.

The intersection is small relative to its parent sets: of the 414 papers judged to involve Design, 222 also involve Grounding, and of the 1,451 papers judged to involve Grounding, the same 222 reach a design. The smallness traces to the parent sets, above all to Design: Agentic Design is a young field, with its own research appearing only from 2023 onward, as Section 2 shows, so its intersection with the much larger Grounding literature starts from a small base. Design Grounding is small because the field it grows out of is small, and it is young: it appears a year after Design-only and

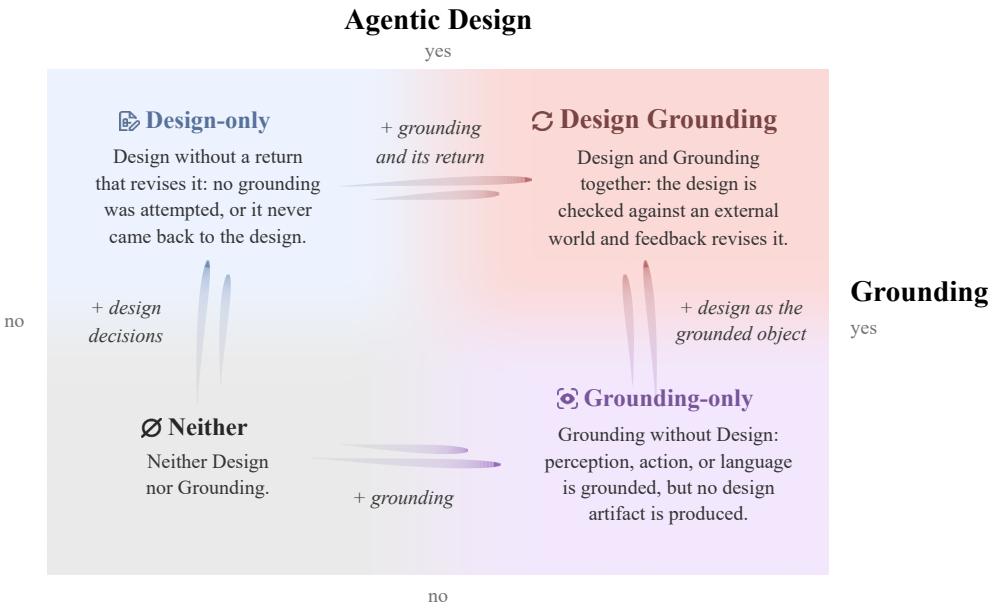


Fig. 5 The four quadrants. The quadrants used in the screening are thresholds over conditions that vary by degree. The vertical axis indicates the presence of Agentic Design and the horizontal axis the presence of Grounding, each judged from the title and abstract; unclear judgments are routed to the full-text stage. Each strand crosses one axis and names the condition that crossing adds; Design Grounding requires both legs of the relation at once, an outbound reach to the External World about the design and a return from it that revises the design. Design-only therefore holds designs that lack either leg, whether no grounding was attempted or the grounding never came back. No strand runs along either diagonal, since those pairs differ by two conditions at once. Strands describe the defining conditions rather than the movement of individual papers, each of which the screening places in one quadrant. The number of papers in each quadrant and their growth over time are reported in Figure 6. The quadrant icons were made by UIcons from Flaticon.

four years after Grounding-only, yet it grows the fastest of the four quadrants, which makes its quality a first-order question.

3.3 The Direct Design Grounding Corpus

Direct marks a difference from grounding work whose feedback reaches an intermediate artifact: in a Direct Design Grounding study, the signal from the External World updates the design itself. The full-text and human stages reduced the corpus to the 20 papers that satisfy the inclusion rule: Design at least L1 and Grounding at least G1. The 20 papers are drawn from arXiv (10), arXiv and DBLP together (6), and DBLP (4); the grey literature contributed no included paper. Table 9 cross-tabulates Design level and Grounding strength, and Table 10 lists the 20 papers individually, with the Design artifact, the External World referent that anchors the Grounding, and the feedback loop that closes it.

Three patterns stand out in this corpus. We report them as observations here, and examine the mechanisms behind them in Section 4.2.1.

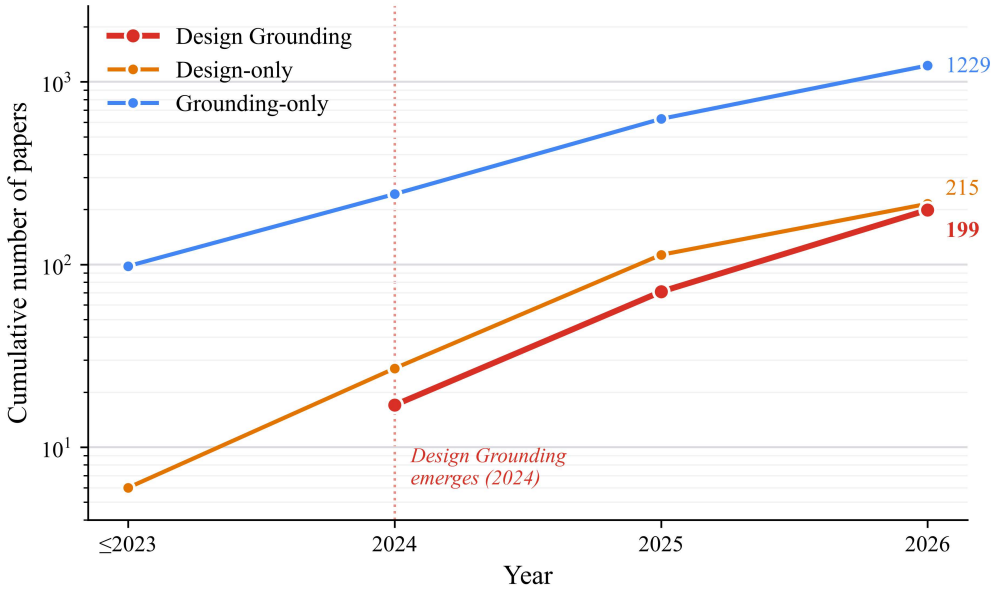


Fig. 6 Growth of the three quadrants. Cumulative counts for the quadrants that involve Design or Grounding, on a logarithmic scale. Design Grounding appears only in 2024, yet by 2026 it has nearly caught up with Design-only, which started a year earlier.

3.3.1 Grounding almost always stops at simulation

Of the 20 papers, 19 close the loop against a model of the system and one against a measurement. The single exception is the cardiac digital twin work of Zhou et al. (2026), in which real clinical electrocardiogram data enters both the training and the agent feedback loop. The other 19 papers, even those that use a real process, stop at the simulated prediction: for example, TSMC 28nm synthesis in Tang et al. (2024) and SkyWater 130nm in Allam et al. (2025) both remain unmeasured, because the chip is never taped out. The gap between G2 and G3 is therefore the difference between a calibrated model's prediction and a measurement of physics. This is the most direct quantitative observation of the Grounding gap: *even the work that closes the loop, closes it against simulation rather than against reality.*

3.3.2 Design from scratch dominates

Thirteen of the 20 papers (65%) are graded L3, where no part of the structure is given and the agent determines it from scratch. Most of them design a complete system. The component-level (L1) and restricted-scale (L2) cases are sparse: four and three papers respectively. The scarcity reflects the research landscape: a closed loop is easier to demonstrate for a complete system with a mature simulator than for a single component whose design space is constrained.

Table 10 The 20 papers in the Direct Design Grounding corpus. L denotes the Design level and G the Grounding strength, as defined in Section 3.1.3. The External World referent is the independently existing world object that anchors the Grounding.

Work	Design artifact	External World referent	Feedback loop	L/G
Ren et al. (2026) Self-evolving coding agents	Aerodynamic surrogate pipeline	CFD (Navier-Stokes)	Evolutionary selection	L3/G2
Wang et al. (2026c) Concrete barrier design	Concrete barrier	Yield-line analysis	Generate-evaluate-optimize	L2/G2
Nallathambi et al. (2026) A3D	Hardware accelerator (HLS)	Catapult HLS synthesis	Synthesis report	L3/G2
Lie et al. (2026) Execution-grounded simulation	Reservoir simulation model	JutulDarcy solver	Execution grounding	L3/G2
Bao et al. (2026) AnalogAgent	Analog circuit	SPICE	Simulation feedback	L3/G2
Zhao et al. (2026) AutoMS	Microstructure	COMSOL EM solver	Simulation search	L3/G2
Wang et al. (2026a) Coding agent as world simulator	Physical world simulation	MuJoCo, Chrono, Isaac Gym	Physics consistency checks	L3/G2
Tian et al. (2026) Multi-agent LLM process design	Chemical process	Process simulator	Convergence feedback	L3/G2
Zhou et al. (2026) Cardiac digital twins	Cardiac digital twin	Real clinical EGM data	Real-data loop	L1/G3
Chen et al. (2026) Photonic inverse design	Photonic device	EM simulation	Design-skill memory	L3/G2
Wang et al. (2026e) MDForge	Molecular dynamics pipeline	MD engine	Sparse simulator feedback	L3/G2
Zhang et al. (2026) PANDA	Analog circuit	SPICE + PEX	Performance-driven iteration	L3/G2
Berger et al. (2026) Physics-in-the-loop CAD	CAD mechanical part	FEA solver	Physics-tool feedback	L2/G2
Wang et al. (2026d) Robot design trials to skills	Robot morphology	EVOGYM physics engine	Evolution and skill reuse	L2/G2
Wang et al. (2025) AnalogSAGE	Analog circuit	SKY130 PDK + ngspice	Grounded experience	L1/G2
Allam et al. (2025) ASIC agent	ASIC	OpenLane + SkyWater 130nm	Synthesis and DRC feedback	L1/G2
Tang et al. (2025) MAHL	Chiplet	Real-process PDK synthesis	Adaptive debugging	L3/G2
Patra et al. (2024) AiEDA	Digital ASIC	EDA toolchain	Tool feedback	L3/G2
Lai et al. (2024) AnalogCoder	Analog circuit	SPICE (ideal device model)	Self-correction	L1/G1
Tang et al. (2024) HiVeGen	Chip (Verilog)	TSMC 28nm synthesis	Synthesis feedback	L3/G2

3.3.3 The corpus concentrates in hardware and scientific computing

Design Grounding does not distribute evenly across the design fields; it concentrates in hardware and scientific computing, the same fields that dominate the Design-only corpus of Section 2. However, the corpus thins out in software engineering, where execution can be tested but the real-world conditions a system must meet have no counterpart of a PDK or a physics engine, and this contrast sharpens the observation. Section 4.2.1 examines the mechanism behind this asymmetry.

3.4 Near Misses and Boundary Patterns

The 155 papers excluded in the manual review are not noise. They cluster into recognizable patterns that show how work in this area almost reaches Design Grounding, and where the connection breaks.

3.4.1 Strong Design with invalid Grounding

The most instructive excluded group is the ten papers with Design at L3 but Grounding at G0. These papers build complete systems from scratch, yet their grounding does not survive the criteria. Three sub-patterns recur.

- *Constructed benchmarks and tools:* the design is evaluated against a benchmark or a toolchain built for evaluation rather than an independently existing world. Examples include a full RISC-V CPU validated only through the OpenROAD/ASAP7 toolchain, where the process design kit is a predictive academic artifact rather than a foundry PDK (Krishna et al. 2026); Verilog evolved against compilation and functional simulation (Yu et al. 2026); databases synthesized against TPC-C and YCSB workloads (Lou et al. 2026; Liu et al. 2026b); and neural architectures evaluated by training loss on constructed tasks (Pepe et al. 2026).
- *Learned or empirical models:* grounding relies on deep-learning predictors or empirical scoring functions rather than physics-law engines, as in protein design grounded in structure predictors and scoring functions (Rahman and Rahman 2026).
- *Human-in-the-loop verification:* the main feedback channel is manual inspection, which violates the autonomous closure requirement, as in a finite element model of a concrete barrier whose geometry is checked by eye (Wang et al. 2026b).

The pattern across all ten is the same: the Design is real, but the world it is checked against is either constructed, learned, or mediated by a human. Section 4.2.2 analyzes the mechanism behind this substitution.

3.4.2 Design as research automation

The largest excluded group, 132 papers, fails at the Design level: the agent is doing research automation rather than system design. These papers optimize parameters, tune configurations, search hyperparameters, or execute experiments. The artifact they produce is a tuned configuration, a set of parameters, or an adjustment to an artifact whose structure is already fixed: the agent decides values inside a structure it did not choose, so it produces no specification of what is to be built. Constructing a computational artifact counts as Design on the same terms as constructing a physical one, and what separates them is whether the agent determines the artifact's structure rather than whether the artifact is a chip, a model, or a program. The line is drawn at the *artifact*: research automation produces decisions about how the work proceeds, while Design produces a specification of what to build. This semantic drift is common enough to warrant explicit attention, because it dilutes the meaning of Agentic Design in the literature.

3.4.3 Closure without problematization

A subtler boundary runs through the 20 included papers themselves: whether the authors treat grounding as a research problem in its own right varies widely. Based on our manual reading of the full texts, we group the papers by this positioning:

- Eight frame grounding explicitly as the subject of the work, for example by calling the simulator the arbiter of physical validity (Lie et al. 2026) or the physical consistency check the source of grounded signals (Wang et al. 2026a).
- Eight treat it as an intermediate mechanism.
- Four use a synthesizer or simulator as a default pipeline step without discussing the grounding it provides (Allam et al. 2025; Tang et al. 2025; Patra et al. 2024; Tang et al. 2024).

Practice closes the loop in all three groups; the difference is whether the closure is the subject of the paper or an incidental tool. This suggests that even where Design Grounding exists in practice, *it is often not problematized as such.*

3.5 Summary

The systematic pipeline yields three results. First, the Design Grounding quadrant is the smallest of the four, at 6.7% of the screened abstracts, and it emerged only in 2024. Second, the 20 papers that survive full-text adjudication and manual review close the loop almost exclusively against simulation: 19 of 20 stop at G1 or G2, and only one reaches real measurement. Third, the near misses are systematic: strong Design frequently pairs with invalid Grounding, and the most common failure among them is that the design is checked against a constructed, learned, or human-mediated world rather than an independently existing one. These results set up the challenges analyzed in Section 4: if the loop rarely closes, and closes against simulation when it does, then the Design level poses specific grounding problems that existing mechanisms do not yet address.

4 Grounding Challenges and Consequences at the Design Level

The screening of Section 3.3 and Section 3.4 raises two questions about the loop it describes: *why it so rarely closes, and what it means that it so rarely closes, given what a design decision governs.* The first concerns difficulty, the second consequence, and this section answers both.

1. **Design Grounding is hard.** The difficulty follows from the structure of the Design object and from the limits of the model that produces it, and it appears as four concrete obstacles along the loop of observation, feedback, revision, and evaluation.
2. **Design Grounding is consequential.** A design error is the most leveraged kind of error, because its cost propagates through everything that follows, and in some domains the error cannot be corrected at all.

Difficulty alone would describe a technical challenge, and consequence alone would describe a risk. The two are also connected, because the same properties that keep the loop from closing are the ones that make the consequences severe. The Introduction presented the two as the case for this survey, and the evidence below is what supports them.

4.1 Why the Gap Is Structural

Grounding pairs an internal representation with something outside it, and any such pairing requires two things to be available: a referent that can be observed (Harnad 1990), and a signal that returns while the decision that produced it can still be identified (Minsky 1961; Pignatelli et al. 2023). The design setting supplies neither in the form that grounding requires. Besides, the model cannot supply it instead: its competence is drawn from language, and language reaches a referent through prior descriptions of it (Coelho Mollo and Millière 2026). The gap is therefore structural in two senses: *two of its causes lie in the Design object, and one lies in the model itself.*

4.1.1 The referent does not exist yet

The referent of a design is a system that has not been built. The world that the design must satisfy exists and can be observed, yet the conditions that decide whether the design will work appear with the realized object, which is available only after the design is complete and implemented. During the design phase the observation channel is therefore largely closed, and the proxies available are predictive models, simulators, or constructions built for the purpose of evaluation. This single fact helps explain two findings of Section 3.3 and Section 3.4: almost all closed loops stop at a simulator's prediction, and the near-miss designs are checked against worlds that are constructed, learned, or mediated by a human. Both are responses to the same absence: *when the real object cannot be observed, the loop substitutes something that can be.*

4.1.2 The signal returns too late

A design decision does not produce an observable consequence by itself; the consequence appears after implementation, manufacturing, and deployment. The feedback signal and the decision that caused it are separated by the entire realization chain, and the separation has two effects. The first is that the design phase offers no corrective pressure: a design that will fail is indistinguishable from one that will hold at design time, so the loop has no reason to reopen until the failure has already been committed. The second is that the returning signal no longer points to its cause. A consequence that arrives after many intervening steps must be traced back through all of them, which makes attribution a problem in its own right rather than a step that follows from observation, and Section 4.2.3 examines it as such.

4.1.3 The model cannot supply the referent

The two causes above concern the object that is designed. A third cause lies deeper, in what the model itself can supply. A design specification is a structure of symbols,

and the question of how such structures acquire contact with the world is the symbol grounding problem (Harnad 1990). Formal work on that problem reaches a limit directly relevant here: a purely symbolic system, and any system whose meanings are fixed in advance, cannot ground itself, because grounding requires a process that is external to the system, dynamic, and not reducible to internal derivation (Liu 2026). Contemporary accounts that allow language models to achieve referential grounding spell out what it takes instead: a representation is referentially grounded when it stands in an appropriate causal-informational relation to its referent and has been selected for that role (Coelho Mollo and Millière 2026). A design that has not been built cannot enter such a relation at design time, because the artifact that would supply the causal contact does not yet exist. The model can therefore produce a fluent and internally coherent design without being able to certify that it will work, and additional scale or training data does not close the difference. *The missing ingredient is contact with a referent that does not yet exist.*

4.2 How the Difficulties Appear Along the Loop

The root causes above are properties of the setting, and they become visible as four concrete difficulties when the design process is read as a closed loop. A useful lens is the *partially observable Markov decision process* (Kaelbling et al. 1998), which reads the Design Grounding loop of Figure 7 as a decision problem under partial observation. Table 11 lists the eight elements of the process and their counterparts in the design setting. The mapping is organizational: it states where each difficulty sits in the loop, and it is not used here to derive new conditions on the design process.

Table 11 The elements of a partially observable Markov decision process and their counterparts in the design setting. Rows that reference a section are examined in that section.

POMDP element	Design Grounding counterpart
State S	The External World, the environment the design must ultimately satisfy, hidden from the agent
Action A	A design decision, a candidate specification of the system to be built
Transition T	The consequences of the design after implementation, manufacturing, and deployment
Reward R	External feedback on design feasibility
Observation Ω	The feedback channels: simulation output, synthesis reports, measurements
Observation function O	How feedback signals map to the true state, carrying the simulation error of Section 4.2.1
Discount γ	The decay of delayed feedback, the feedback delay of Section 4.1
Initial belief b_0	The agent's prior assumptions about the world, the origin of the Internal-to-External Bias

The four subsections that follow trace this loop in order: observation, the authority of returning feedback, revision, and evaluation, and finally with cross-domain transfer as the closing case. Each difficulty traces back to a root cause of Section 4.1.

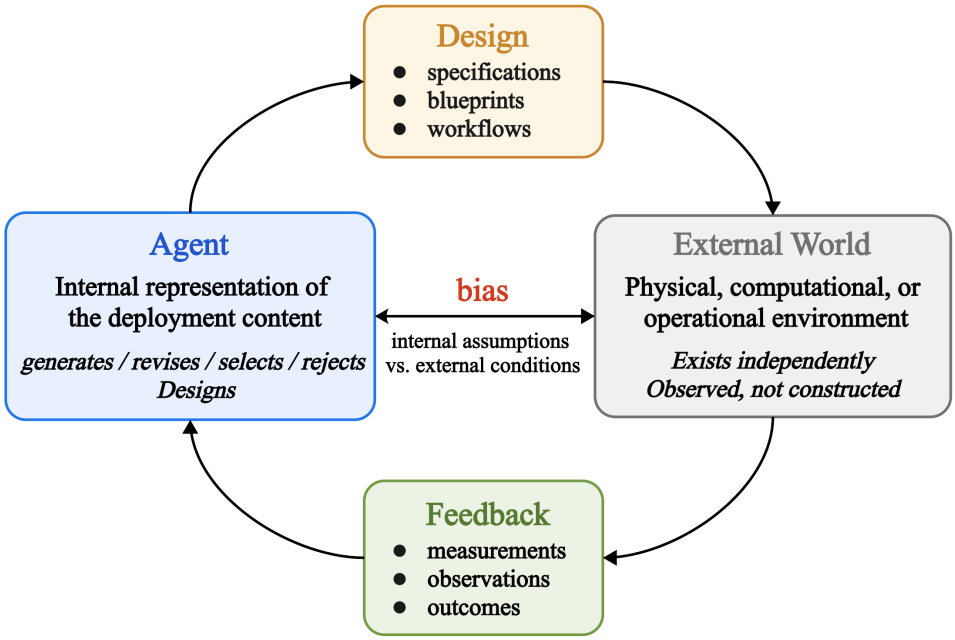


Fig. 7 The Design Grounding loop. An agent generates a Design, such as a specification, blueprint, or workflow, based on its internal representation of the deployment context. The Design enters the External World, the physical, computational, or operational environment that exists independently of the agent and is not constructed for evaluation. Observable referents in that world expose the bias between the agent's internal assumptions and the external conditions the design must satisfy. Feedback, in the form of measurements, observations, or outcomes, returns to the agent, which revises, selects, rejects, or terminates the Design. The loop closes only when external evidence changes the design itself. The four challenges analyzed in Section 4 concern, in order, observing the design before it exists, trusting the feedback that returns, revising the design from that feedback, and measuring the outcome.

4.2.1 Simulation as the Default Observation Channel

The difficulty. The design object does not exist during the design phase, so the only observation channel available is simulation, with a fidelity ceiling at the prediction of physics.

Section 3.3 reports that 19 of the 20 papers close the loop against a physics-law engine, a simulator, or a synthesis tool, and only one reaches real measurement. This is a structural fact, not a preference: the physics-law engine is the interface to the External World that is available at design time, before the object exists. A synthesizer, a finite-element solver, or a process design kit embeds the relevant physics in a computable form, which makes the world queryable before it exists. The same availability underlies the design scope of Section 3.3.2: a complete system has a mature simulator that covers its component interactions, while a single component has no such coverage, so a closed loop is easier to demonstrate at the system level.

The channel is real, but lossy at several points. The model approximates the physics through discretization and simplifying assumptions, and its parameters are calibrated on a reference line, so a process design kit predicts a process that differs from the actual foundry line. And the transfer from simulation to the deployed setting is never perfect, a gap that robotics has quantified (Collins et al. 2020) and surveyed (Zhao et al. 2020). The corpus shows the consequence: even the papers that synthesize against a real process, TSMC 28nm in HiVeGen (Tang et al. 2024) and SkyWater 130nm in ASIC-Agent (Allam et al. 2025), stop at the simulated prediction, because the chip is never taped out. The predictive academic process design kit behind a near-miss paper, the ASAP7 kit of the RISC-V design of Section 3.4 (Krishna et al. 2026), sits one step further back: it models a process that is not a foundry line at all, so its predictions are anchored to no measured process.

Across the corpus, the loop stops at a model of the system in 19 of the 20 papers, and G3, the measurement of physics, has been reached once, in the cardiac digital twin work (Zhou et al. 2026). Reaching G3 requires the realized object, and realization happens after the design decision, so the design phase rarely provides the conditions for it. Therefore, the gap between G2 and G3 is not a matter of effort. It is the boundary of *what an observation channel can deliver before the object exists*, and it helps explain the hardware-software contrast of Section 3.3.3: hardware and scientific computing have such channels, while software engineering does not. In software engineering, the properties that determine whether a system works, such as user behavior, deployment environment, and operational load, have no direct equivalent of a PDK or a physics engine, so the field is left without an observation channel of this kind.

In POMDP terms. The simulator is the observation function, so the agent conditions its policy on a predicted state rather than on the state itself. The fidelity of that function bounds what any policy can achieve, which is why more search cannot substitute for a better channel.

The channel has a second weakness, beyond its fidelity. It can be constructed by the agent itself, which raises the question of whether the signal that returns is grounded at all.

4.2.2 The Authority of Feedback

The difficulty. The evaluation world can be constructed by the same party that produces the design, which strips the feedback of its independence.

Section 3.4 reports the most instructive excluded pattern: strong designs evaluated against worlds that are constructed, learned, or mediated by a human. The three sub-patterns, constructed benchmarks and tools, learned or empirical models, and human-in-the-loop verification, share one mechanism. In each, the evaluation logic is controlled by the same party that produces the design, and the External World loses the independence that the loop requires. Grounding then degenerates into an internal consistency check: the design is checked against a world that the agent, or the research team, constructed for the purpose of evaluation.

The degradation is a known failure mode. Specification gaming describes how agents exploit the difference between the stated objective and the intended one (Krakovna et al. 2020), and it persists in modern reasoning models (Bondarenko

et al. 2025). Reward misspecification shows that the gap between the proxy and the target is systematic, not incidental (Pan et al. 2022). The design-level version is subtler, because the proxy is an entire evaluation world rather than a scalar reward function. When the world is constructed, the agent optimizes against the construction, and the bias that Design Grounding is meant to expose, the divergence between the internal judgment and the external realizability, becomes invisible by construction.

The distinction between an external tool and a constructed evaluator matters here. A synthesizer or a simulator is external to the agent, and its error, described in Section 4.2.1, is the error of a physical model. By contrast, a benchmark, a learned predictor, or a testbench written by the agent itself carries that same model error plus the agent's own preference. The two errors are different in kind: the first can be characterized and bounded, while the second is self-referential, and evaluators built on the same model that produced the output have been shown to favor their own generations (Panickssery et al. 2024). This is why a design checked against a physics-law engine and a design checked against a constructed benchmark are not the same failure. The first is limited by fidelity, and the second by independence.

The authority of feedback therefore has two preconditions that practice often violates: *the feedback must come from a source outside the agent, and that source must not be constructed for the purpose of evaluation.* When either precondition fails, the loop appears closed but measures internal consistency instead of external correspondence. This explains why the near-miss designs look like Design Grounding to their authors while failing the criteria of Section 3.1.3.

In POMDP terms. The observation function is supplied by the agent. When the same party chooses the feedback and produces the design, the belief update is conditioned on a function the agent controls, and the posterior can converge on the initial belief whatever the state does.

Even when the feedback is authoritative, however, it has to change the design, and that step carries its own difficulty.

4.2.3 From Feedback to Structural Revision

The difficulty. Feedback that cannot change the design does not ground it. Today's feedback drives parameter-level revision, and it rarely reaches the structure.

The depth of revision that a feedback signal can drive depends on the structure of the design space, and the difficulty of revision is the next structural constraint after the authority of feedback.

The mechanisms that turn feedback into updates, gradient descent, policy gradient, evolutionary selection, assume a space in which a small change produces a small effect. However, a design space is different in kind: the topology, the component choices, the interfaces, and the parameter values are not uniformly continuous, and a structural change is a discrete jump rather than an incremental adjustment. The corpus also reflects this limit. The feedback mechanisms in Table 10 are mostly iterative refinement within a fixed architecture, for example synthesis reports (Nallathambi et al. 2026), simulation feedback (Bao et al. 2026), and performance-driven iteration (Zhang et al. 2026). *Few of the 20 papers report a structural revision driven by the feedback*: two evolve designs against measured performance and carry the result forward as reusable

knowledge (Ren et al. 2026; Wang et al. 2026d). That route buys structure with evaluations rather than with information, and evaluations are the resource the design setting cannot spare, as Section 4.3 argues. The design spaces they operate in, circuit topologies, chiplet partitions, accelerator architectures, are spaces where the gradient assumption fails.

The second difficulty is attribution. A failure signal arrives from the whole system, and it must be traced to the design decision that caused it. The realization chain described in Section 4.1 separates the decision from the consequence, so the attribution is not a local problem. It requires a causal model of how the design maps to system behavior, and that model is exactly what the agent lacks when the design is novel. This is the mechanism behind the most frequent difficulty reported by the Design-only corpus in Table 3: the agent can recognize that the design fails, and it can describe the failure, but it cannot locate the failure in the design, because locating requires credit assignment across the realization chain.

The revision step therefore has a depth gradient.

- *Parameter revision* is reachable with current mechanisms, because the feedback is locally informative.
- *Component-level revision* is harder, because it requires knowing which component to replace.
- *Structural revision* is hardest, because it changes the design's architecture rather than its values, and it requires a causal account of the whole.

The corpus stops at the first two steps apart from the two evolutionary searches that buy structural change with evaluations rather than with information, and the difficulty evidence of the Design-only corpus suggests why: in the design setting, the limitation is an attribution problem, the inability to trace a failure to the design decision that caused it.

In POMDP terms. Attribution is credit assignment under partial observation. A single reward has to be traced back to one of many discrete actions, and the transition from action to outcome crosses implementation, manufacturing, and deployment, which leaves the assignment underdetermined.

Revision, when it happens, must also be verified, and verification is the last stage of the loop.

4.2.4 What Current Evaluation Measures

The difficulty. Evaluation measures the final artifact rather than the correspondence between the artifact and the world it will meet.

Nearly all evaluation in the 20 papers takes the form of task scores, simulation metrics, or artifact quality. And the same dominance holds at the field level: evaluation of LLM-based agents is dominated by end-task performance on static benchmarks (Yehudai et al. 2025). For Design Grounding, this is a measurement mismatch, because the quantity of interest is the correspondence between the artifact and the world it will meet, and the artifact score captures only one side of it.

The mismatch has two parts. First, the final score is insensitive to the Internal-to-External Bias. A design that is internally coherent but externally infeasible can

score well, because the benchmark measures the design against a constructed setting rather than against the conditions that the design will actually face. This is the same constructed-world problem as in Section 4.2.2, at the level of measurement rather than feedback. Second, the correspondence itself is a process property: it is visible only through the trajectory of revisions, the moments where the bias is exposed, and the corrections that follow. Yet static benchmarks record none of this.

The absence of process-level measurement has a consequence for the field itself. Section 3.4 observes that even where the loop closes, the closure is rarely the subject of the research. Part of the explanation is evaluative: when no measurement exposes the bias, the bias does not become a research problem. The papers that do problematize grounding, a minority of the corpus, are also the ones that measure something process-like, such as the physical consistency of the simulation against the world model (Wang et al. 2026a) or the authority of the simulator as arbiter of validity (Lie et al. 2026). The evaluation mismatch and the closure-without-problematization pattern are therefore the same phenomenon seen from two sides.

In POMDP terms. The reward is defined over the artifact. The score evaluates the action itself, while the quantity at stake is whether the design's assumptions about the state survive contact with that state.

The four difficulties of this section, the limited observation channel, the fragile authority of feedback, the bounded depth of revision, and the mismatched evaluation, are structural in the sense that they follow from the root causes described in Section 4.1 and from the feedback arrangements those causes force. Whether they are specific to the design level, or whether existing grounding mechanisms can be adapted, is the question of the next subsection.

4.2.5 Cross-Domain Structure and Transfer Boundaries

The structure of the previous four subsections is not specific to any one domain. It recurs wherever an agent makes design decisions without direct access to the environment those decisions will encounter, which is why the landscape of Section 3 places Design Grounding at the intersection of literatures that developed in isolation.

The first axis of comparison is the feedback infrastructure gradient. Hardware has a process design kit, a synthesizer, and a simulator, and scientific computing has physics-law solvers, so the observation channel of Section 4.2.1 exists and is queryable. Software and interactive systems have no equivalent, as Section 3.3.3 notes, so the same mechanisms are unavailable. This gradient maps where the observation channel exists rather than ranking the fields themselves, and it predicts both where the corpus concentrates and where grounding is hardest.

The second axis is the traditional grounding mechanisms discussed in Section 2. Symbol grounding pairs an internal symbol with an observable referent (Harnad 1990), perceptual grounding pairs a state estimate with a current observation, and action grounding, as in embodied agents, pairs a policy with the consequences of executing it (Ahn et al. 2022). Table 12 lists the presupposition each mechanism relies on and where it fails at the design level.

Table 12 Traditional grounding mechanisms, the presupposition each relies on, and its failure at the design level.

Mechanism	Presupposition	Failure at the design level
Symbol grounding	The referent exists and can be pointed at	The design does not exist yet
Perceptual grounding	The current state is accessible	The state of the future system is not accessible
Action grounding	Execution is cheap enough to repeat	Implementation and manufacturing are too expensive to repeat

At the design level, each presupposition fails in the way that Table 12 summarizes: the first two failures correspond to the causes of Section 4.1, and the third to the cost examined in Section 4.3.

Given both axes, the transfer boundary follows. Robotics has developed the most mature route from simulation to reality, through domain randomization and systematic quantification of the reality gap (Tobin et al. 2017; Zhao et al. 2020). That route works because the real world exists during training and can be sampled, even if sparsely. However, the real world does not exist during the design phase, so the sim-to-real route applies only after a candidate design is realized, which is precisely the step that the agent cannot perform. The same boundary applies to the other traditions. Grounded language and symbolic methods require a referent that is available (Harnad 1990), and verification-based approaches in software engineering require a deterministic oracle, which the properties visible only in deployment do not provide. The mechanisms transfer only up to the boundary where the object appears, and the boundary is structural.

This returns the discussion to the renovation scenario of Section 1: from the drawing board to the site, and back. The scenario is the same loop, with the same structural causes, in a setting where the site exists and can be measured. The agentic version of the scenario, however, is harder in every element. The renovation metaphor appears simple, a drawing, a site, and a revision, yet it already carries the full structure of Design Grounding: the drawing is a Design, the site is an External World that exists independently of the agent, the measurements expose the Internal-to-External Bias, and the revised drawing is the design revision. In the agentic setting, each element degrades: the site is a simulation or a constructed world, the measurements are delayed or absent, and the cost of rework is carried by whoever commits the design.

One loop, four failures, and then one cost. Read along the axis of Figure 8, the four are one failure seen at four points: at each stage the loop keeps its shape and loses its reference, so a design can pass every check the loop offers and still meet a world it was never tested against. What separates them is when the failure can be caught, and the earlier a stage sits, the more of the final cost its decisions govern and the less likely anything is to measure it. The band that widens to the right is that asymmetry: the design freeze fixes the error in place, and correcting it afterwards means undoing work already done along the realization chain. Section 4.3 puts that cost in numbers.

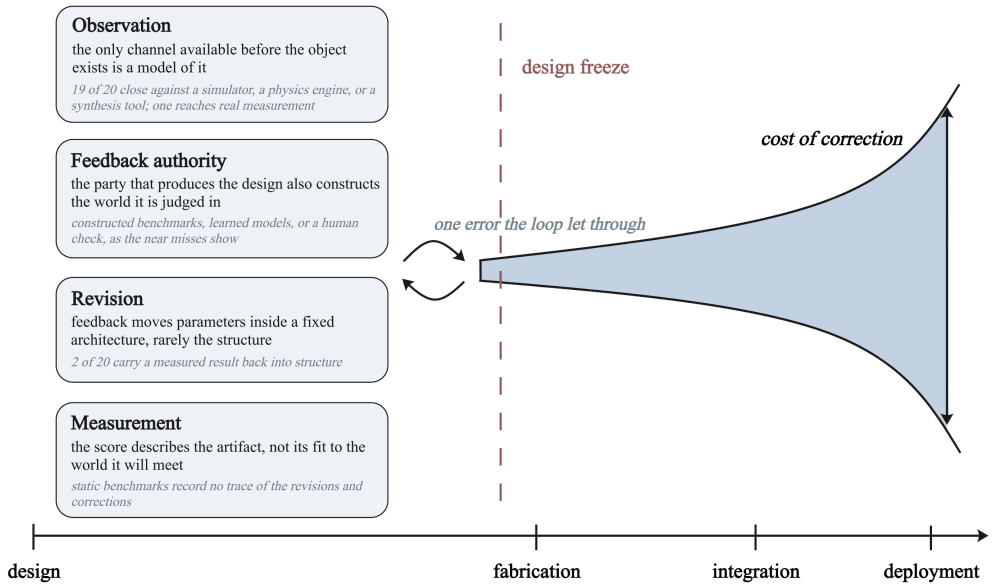


Fig. 8 One error through the loop, and the cost that follows the freeze. The four boxes name the difficulties of this subsection and the form each takes in the corpus. The loop lets one error through, the design freeze fixes it in place, and the band that widens to the right is what correcting it costs along the realization chain.

4.3 What the Gap Costs

A failed design carries consequences paid downstream in fabrication, integration, and rework. Design therefore cannot be treated as a trial-and-error process in the way that robot learning can, because the cost of an experiment is orders of magnitude higher and the outcome is not easily reversible. This asymmetry makes the most direct grounding strategy, run many designs and observe which one survives, largely unavailable, and it is one reason why the trial-and-error mechanisms discussed in Section 4.2.5 do not transfer to the design level.

The stakes follow from this asymmetry. Design decisions carry the greatest leverage over what a system ultimately costs. In a study of industrial unmanned aerial vehicle development, decisions made before design freeze accounted for 86% of the total expected program cost, 34% of it before concept freeze, and a design defect detected late carried a rework multiplier of more than ten (Tan et al. 2017). The consequence is structural as much as empirical: an error that survives design is not repaired where it was made, and correcting it later means undoing work that is already done.

Irreversibility sharpens the picture. A design error is recoverable when the artifact can be rebuilt, and the papers that close the loop against a real process show how rarely that condition holds (Tang et al. 2024; Allam et al. 2025). Where realization consumes a physical experiment, a wrong design is not corrected but paid for. The harm is not only wasted effort: a design that satisfies every internal check and fails

in the world carries that failure past the point where anyone can trace it back to the decision that caused it. The consequences establish why Design Grounding matters, and they do so without appeal to how much attention the problem has received.

4.4 Summary

The Design level poses grounding problems that the mechanisms of the existing literatures do not yet address, for reasons that lie in the object and in the model. The referent of a design does not exist yet, and the model that produces the design cannot supply it, so the observation channel is largely closed and simulation becomes the default, with a fidelity bound at the prediction of physics. The signal that returns is delayed, so attribution across the realization chain is hard and revision stops at the parameter and component level. Those causes appear as four difficulties along the loop: a substitute observation channel, a fragile authority of feedback when the evaluation world can be constructed by the agent, a revision that rarely reaches the structure, and an evaluation that measures the artifact rather than its correspondence with the External World. The stakes are correspondingly high, since design decisions carry the greatest leverage over what a system costs and the cost of correction grows along the realization chain. These mechanisms account for the patterns that Section 3 reports, from the rarity of closed loops to the constructed worlds of the near misses. The difficulty and the consequences run together: the properties that keep the loop from closing are the same ones that make a design error costly. These four difficulties define the open questions examined in the next section, and the cost of the gap is what makes those questions worth answering.

5 Open Problems and Research Directions

The four difficulties of Section 4, the limited observation channel, the fragile authority of feedback, the bounded depth of revision, and the mismatched evaluation, set the agenda for this section. For each, we derive the properties a solution would need, point to the nearest existing work in adjacent fields, and mark where the gap remains. Section 5.1 takes observation, Section 5.2 the authority of feedback, Section 5.3 revision depth, Section 5.4 evaluation, and Section 5.5 asks what it would take to hold the four together. Figure 9 states the position the field is in: four candidate parts already exist in some form, each in a different field, and no field has assembled them into one loop. Which further parts the loop needs is part of what remains open, and the four named here are a starting set. Table 13 then sets out, for each direction, what would make it feasible, why it is urgent, and what a result would have to show.

5.1 From Simulation to Grounded Observation

The difficulty. The observation channel is closed before the object exists, with a fidelity ceiling at the prediction of physics (Section 4.2.1).

Open question. Can the observation channel approach real measurement before the object exists, or at least make the uncertainty of the simulation a quantifiable quantity?

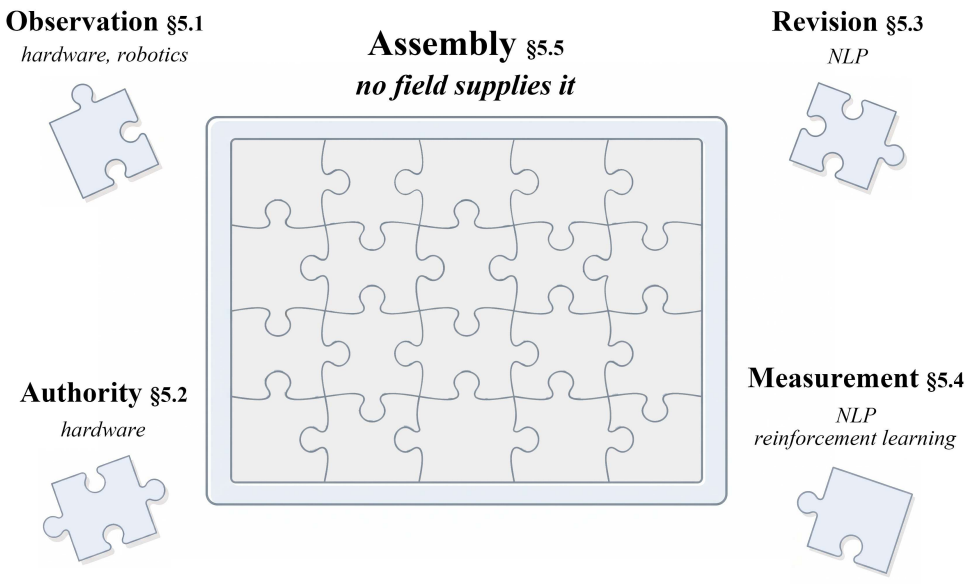


Fig. 9 Four candidates for one assembly. Each loose piece is one candidate part that a solution would contain, paired with the field that already supplies something like it: differentiable simulation and sim-to-real measurement for observation, foundry-released kits for authority, language-model search for revision, and benchmark governance for measurement. The board is the assembly, its sockets stay empty, and no field installs even the four parts it has. The board illustration was created with an AI image generation tool.

Directions. The primary direction that does not depend on the realized object is to change the simulation itself. A physics-law engine that is differentiable turns the observation channel into a gradient field over the design space, so the agent can query both whether a design is feasible and in which direction it becomes more feasible. Differentiable simulation has already been applied to high-dimensional inverse design, where it outperforms black-box optimization over strongly coupled design parameters (Crilly et al. 2026). This direction connects directly to the revision problem of Section 4.2.3: a differentiable channel supplies the continuous gradient that the design space otherwise lacks.

The other two directions require real measurement, and therefore work only when some real data already exists. Multi-fidelity optimization calibrates a cheap model with a small number of expensive high-fidelity evaluations, which is a mature technique in engineering design (Do and Zhang 2023). In Design Grounding, however, the high-fidelity evaluation is the measurement of a realized object, and realization happens after the design decision, as Section 4.1 shows. The same boundary applies to active sampling, which turns a limited measurement budget into a targeted experiment (Xia et al. 2025). Both directions therefore describe the iteration from one generation of designs to the next, where measurements of the previous generation enter the loop, rather than the first design of a new artifact.

The maturity of the three directions follows the same order. Differentiable simulation is the path available at design time without a realized object, yet its surrogate error is a new simulation error of the kind described in Section 4.2.1. Multi-fidelity and active sampling are mature in their home domains but presuppose exactly the real measurements that the design phase lacks.

5.2 Establishing the Authority of Feedback

The difficulty. The authority of feedback is not independent of the designer (Section 4.2.2).

Open question. How can a constructed evaluation world be detected, and how can evaluation be made independent of the designer?

Directions. The most reliable arrangement is institutional rather than algorithmic: the authority to validate a design belongs to a party that does not produce the design. This is the separation of concerns (Dijkstra 1982) applied to the grounding loop, and it has precedents in practice. As with a process design kit, which is released and maintained by the foundry rather than by the designers who use it, the characterization data is an authority that the designer cannot rewrite. The same principle can be realized at the agent level by revoking the agent's control over validation logic and routing it through an external, deterministic evaluator.

The algorithmic complement is detection. Evaluators built on the same model that produced the output have been shown to favor their own generations (Panickssery et al. 2024), which gives a measurable signature of a constructed evaluation world: if the evaluator cannot distinguish its own output from an independent alternative, the loop is measuring internal consistency. A second detection strategy follows: the holdout evaluation world, a setting that the designer does not know and therefore cannot optimize against.

A third candidate is provability: replacing part of the external world with a formal specification that the design must satisfy. The viable target here is the subset of the external world that is formalizable, such as process design rules, physical laws, and safety constraints, which are descriptive statements about the world and can be audited independently. The limiting factor is the specification itself: formal verification depends on one, and specification mining remains a manual, error-prone step (Neider and Roy 2025), which returns the question of who constructs the authority to the specification author. Human judgment cannot fill this gap directly, because an agentic design process closes the loop without a human inside it by definition. Human authority can therefore enter only as an audit of the evaluation world, outside the loop, rather than as a step within it.

5.3 Deepening the Revision Signal

The difficulty. Feedback drives parameter-level revision but rarely reaches the structure, because the design space is discrete and failure attribution crosses the realization chain (Section 4.2.3).

Open question. How can a feedback signal drive structural revision, and how can a failure be traced to the design decision that caused it?

Directions. The first direction is to change the encoding of the design space. If the space is presented to the agent as language, graphs, or component libraries, structural changes become operations on that encoding, and search can proceed combinatorially rather than through gradients. Large language models have been applied to combinatorial optimization as a systematic line of research (Da Ros et al. 2025), and program search with language models has produced mathematical discoveries that eluded classical solvers (Romera-Paredes et al. 2024). The differentiable encoding of Section 5.1 is the continuous complement: it gives the same space a gradient that the discrete encoding lacks.

The other direction is attribution. A failure signal arrives from the whole system, and locating it in the design requires credit assignment across the realization chain. Reinforcement learning for language models faces the same problem at the level of actions within a trajectory, where sparse outcome rewards leave the causing step unidentified (Zhang 2026), and interventional methods have converted failed agent traces into counterfactual repairs, alternative actions that would have avoided the failure (Bonagiri et al. 2026). These seeds operate at the granularity of tokens and steps, while Design Grounding needs attribution from a system-level behavior to a design decision made before implementation. The granularity gap is the difference between the two settings, and it marks where the transfer stops.

5.4 Measuring the Grounding Process

The difficulty. Evaluation measures the artifact, while the quantity of interest is a property of the process (Section 4.2.4).

Open question. What metric can capture that correspondence, and how can process-level evidence be standardized?

Directions. The first direction is process metrics that capture the moments where the Internal-to-External Bias is exposed, the revisions that follow, and the consistency between the simulation and the realized behavior. The reality gap between simulation and deployment has been quantified (Collins et al. 2020), which gives the correspondence a quantitative anchor. Process supervision provides the evaluation pattern: reasoning agents improve when feedback is given at intermediate steps rather than only at the final outcome (Luo et al. 2024), and benchmarks exist that verify the correctness of intermediate steps in multi-tool trajectories (Nath et al. 2025). Adapting this pattern to Design Grounding means scoring the trajectory of design revisions rather than the final artifact.

The complementary direction is grounding-aware benchmarks: constructed settings that expose the bias, rather than settings that reward an internally coherent design. Note that this use of construction differs from the constructed evaluation worlds of Section 3.4. There, a surrogate world stands in for the real one and is presented as if it were the authority. But here, the constructed setting is an instrument: it is designed to provoke the bias, and its purpose is measurement, not validation. A design that survives such a setting has demonstrated its correspondence with the world that the setting samples.

No standardized metric exists, and the process metrics of adjacent fields do not transfer directly, because they assume observable intermediate states in a fixed environment, which the design setting does not provide.

5.5 Grounding Infrastructures across Fields

The difficulty. Each adjacent field has built a grounding infrastructure of its own, and none of them assembles the four stages of the loop into a design-level framework. Section 1 notes that the design and grounding literatures have developed in relative isolation, and Section 2 shows them growing in parallel rather than toward each other. The corpus analysis of Section 3 and the cross-domain comparison of Section 4.2.5 show what this isolation has produced.

The infrastructures. The field-level infrastructures are the highest form of the per-stage solutions discussed above. In hardware, the process design kit, the synthesis and simulation toolchain, and foundry-provided characterization form a validation chain that spans the design-to-fabrication process. In robotics, the sim-to-real pipeline is a staged validation chain, from domain randomization to reality-gap quantification to deployment testing (Zhao et al. 2020; Collins et al. 2020). In software engineering, continuous integration and automated test suites make validation repeatable and auditable. In NLP and reinforcement learning, benchmark governance, dataset releases, and independently maintained leaderboards keep the evaluation world external to the models that are evaluated. Each infrastructure works inside its own field, and each covers only a fragment of the grounding loop.

Design Grounding has no equivalent assembly. The corpus makes the fragmentation visible: each of the 20 papers closes the loop almost entirely through simulation feedback, as Table 10 records, and none combines an authoritative evaluation world with structural revision and process-level measurement. Meanwhile, the Design Grounding quadrant is the smallest of the four, at 6.7% of the screened abstracts. The scarcity is therefore a scarcity of assembly, of the framework that would let a single agent observe, validate, revise, and measure within one design process, while the per-stage methods exist in abundance across the adjacent fields.

The same scarcity bears on the hypothesis left open in Section 2.4, whether feedback-rich iteration could absorb the design stage: iteration absorbs design work largely where observation is cheap and independent of the agent, which is what makes the short loops of action grounding possible, and an evaluation world the agent itself controls supplies neither. Whether the design stage stays occupied therefore turns on the institutional question of the medium term, who supplies the evaluation the loop is judged against.

The renovation scenario of Section 1 returns here. The drawing, the site measurement, the revision, and the verification are separate steps that a renovation firm coordinates as one process. For agentic design, each adjacent field has developed a strong version of one of these steps. No one has built the coordination for an agent that designs. The assembly itself is the open problem, and it is the problem that the per-stage directions of this section cannot solve alone.

5.6 Summary

The directions of this section map onto the difficulties of Section 4 in order: differentiable and calibrated observation for the closed channel, institutional and detection-based authority for the constructed evaluation world, structural encodings and attribution for the shallow revision, and process metrics for the mismatched evaluation.

Table 13 The five directions read against the evidence for them. Feasibility names the nearest demonstration in an adjacent field, urgency names what stays broken without the direction, and the quality bar names what a solution would have to show.

Direction	Feasibility	Urgency	Quality bar
Observation (Section 5.1)	Differentiable simulation already outperforms black-box optimization over strongly coupled design parameters	Every decision taken above the prediction of physics carries an error the loop cannot see	How far a design is from feasible, not only whether it is
Authority (Section 5.2)	A foundry kit already separates the validator from the designer, and self-preference gives a signature to detect	Independence cannot come from inside a loop the designer owns	A designer that cannot reconstruct the evaluation world it is judged against
Revision (Section 5.3)	Language-model search has produced results classical solvers did not, and counterfactual repair exists at step granularity	A parameter-level signal cannot reach the decisions that fix structure	A revision that changes what the design is, and names the decision at fault
Measurement (Section 5.4)	The reality gap has been quantified, and step-level feedback improves agents	Artifact scores cannot separate a checked design from an unchecked one	A count of the quadrant by outcome rather than by abstract
Assembly (Section 5.5)	Every stage has a working infrastructure in some field	The smallest quadrant is the one that needs the most assembly	An agent that observes, validates, revises, and measures within one process

Table 13 sets the five directions against what would make each worth pursuing, and the pattern is uneven in three ways. Feasibility tracks assembly: it is highest where parts already work elsewhere, as in observation and in the cross-field infrastructures, and lowest for measurement, where no standard exists. Urgency runs the other way, since the thinnest support sits under two of the most consequential directions: artifact scores cannot separate a checked design from an unchecked one, and no field assembles the four stages. The quality bar moves the test from the artifact to the process in every case, and it is that shift rather than any single technique that the directions share: the per-stage solutions exist, the field-level infrastructures exist, and no framework combines them for the design setting. This is why the Design Grounding quadrant remains the smallest of the four, and why assembly rather than any further per-stage method is the work that future design agents point toward. The four parts named here are one reading of what the assembly needs, and any other reading leaves the sockets just as empty.

6 Discussion

6.1 The Autonomy Assumption

The first design choice worth discussing is the autonomy requirement: grounding must close without a human inside the loop. This requirement follows from the object of the survey, agentic design systems, whose definitional property is that the design process runs without human supervision. Human-in-the-loop validation is nonetheless among the strongest forms of grounding, and much real design relies on it. The near misses of Section 3.4 include a concrete barrier design whose geometry is checked by eye (Wang et al. 2026b), a pattern excluded only because the check is manual.

Even in such mixed settings, human-in-the-loop grounding faces two structural limits. First, humans cannot anticipate every environment state that a design must satisfy, and Section 4.1 shows that the design setting makes the external world effectively unenumerable. Second, design proceeds through repeated revisions, and each human check adds a manual step to every iteration. The cost grows linearly with the number of revisions, which is incompatible with the high-frequency iteration that an agentic process is meant to enable. The autonomy requirement therefore repositions the human as an auditor of the evaluation world, outside the loop, rather than as a step within it, the same conclusion that Section 5.2 reaches from the definition of an agentic process.

6.2 The Definitions of Design and Grounding

Two definitions introduced in Section 1 determine what the survey can see, and both were sharpened through prompt iteration rather than fixed in advance. The iterations of Section 3.1.2 show the convergence: the Design reading settled on the specification of what is to be built, separated from research automation. The broad Grounding reading failed because calibration and verification loops over actions, skills, or plans were mistaken for grounding over designs, and the design setting required feedback anchored in an External World reference that directly updates the specification. The four-quadrant structure and the L/G grading scheme of Section 3.1.3 are projections of these two definitions rather than independent choices. A different definition would draw a different map, and the framework should be read as a lens on the field rather than a taxonomy of it.

6.3 What Is Design?

The definitions above concern the survey's own lens, and the corpus raises a further question, this time about the field itself. Agentic Design research is active and growing, and it is the origin of this survey, yet the papers screened here rarely revisit **what design is**. Their design objects, accelerators, protocols, scientific workflows, are inherited from human practice and taken as given: the agent optimizes within them, but the objects themselves are not questioned. Whether an agent can redesign the design task itself, break an artifact into different parts, or ask at a different granularity, is a question the corpus does not engage. That silence became a finding of this survey

rather than a premise: *the survey went looking for design agents and found that design itself is seldom the object of design.*

The absence matters to grounding because the grounding problem changes shape with the design object: what can be observed, what can be revised, and how soon feedback returns all depend on what is designed, and rethinking the object is therefore as much a grounding direction as an Agentic Design direction.

6.4 The Breadth of the Inclusion Criteria

The grading scheme of Section 3.1.3 is an analytic construction: Design and Grounding are each graded on four levels, and no precedent exists for either scale. The breadth of the inclusion rule, Design at least L1 and Grounding at least G1, is a deliberate choice and deserves scrutiny. The alternative that we initially considered was the strictest rule, Design at least L3 and Grounding at least G3. However, under that rule, the corpus would be empty: the cross-tabulation of Table 9 records no paper that both designs from scratch and measures the result. Design Grounding would be invisible as a phenomenon. The loose rule was therefore chosen to keep the phenomenon observable, and its cost is boundary ambiguity: the line between L1 component composition and L0 parameter optimization, and the line between an idealized model at G1 and a calibrated one at G2, are drawn by judgment.

The benefit is visible in the corpus itself. The corpus holds both a paper that grounds on a generic model and a paper that grounds on real clinical data, and both are component-level designs. Without them, every closed loop in the corpus would rest on a calibrated model, and the corpus could not separate a design constraint from an availability constraint. The boundary ambiguity has a bounded effect: the mechanism analysis of Section 4 draws its evidence from designs built from scratch, and the component-level papers that appear there are cited for observations about grounding rather than for the mechanisms.

Sensitivity analysis. The two margins of the inclusion rule can be moved in opposite directions, and the corpus responds differently to each: the grade threshold can be tightened, and the autonomy requirement can be relaxed. Most of the filtering is done by the floor of the rule rather than by its height: 132 of the 155 papers excluded in the manual review are L0, and they fall outside the corpus on the Design gate alone. Above that floor the corpus is stable at the design boundary, since requiring L2 instead of L1 removes 4 papers. The grounding boundary is where the corpus is thin: requiring a measurement instead of a model leaves a single paper. That is what the loose rule buys.

The other margin can be relaxed in two ways, and the corpus responds differently to each. The first is the autonomy requirement, which excludes designs whose validation involves a person. Six of the 155 excluded papers mention human involvement in validation; four of them fail the Design gate regardless, one fails an additional grounding condition, and one, the concrete barrier whose finite-element geometry is checked by eye (Wang et al. 2026b), fails on human verification alone. Accepting human verification would therefore add one paper to the corpus. Human involvement is nonetheless widespread in practice, since 33 of the 184 Design-only papers report

some form of real-world validation and 7 report expert dependence as their main difficulty (Section 2.3); those papers sit outside the corpus because their loop does not close. The second direction extends what counts as a system: a computational artifact counts as an engineered system, so an agent that determines the structure of a simulation model or a pipeline satisfies the Design condition on the same terms as an agent that determines the structure of a chip. This extension does not widen the corpus. Of the 37 papers excluded at the Design gate whose grounding would have qualified, each decides values inside a structure it did not choose, such as the sizing of a transistor given a topology or the exploration of a directive space, so none of them enters.

6.5 Mapping rather than arguing

The survey maps the intersection of Agentic Design and Grounding rather than arguing for a solution directly: what the field lacks is a shared account of the problem, and the size of that problem is part of the case for treating it as one. The choice bounds what the survey can claim. It counts and characterizes the gap, and it does not test a design-loop architecture of its own, formalize Design Grounding beyond the internal-to-external level explored here, or prescribe how the loop should close.

6.6 Threats to Validity

6.6.1 Fixed search window and bounded coverage

The search window is fixed and the search space is bounded. The queries cover the period from January 2020 to July 2026, and any work appearing after the window is outside the corpus.

Because Design Grounding is a young area, much of its work appears as preprints and grey literature before formal publication, so the search deliberately includes arXiv preprints and grey literature alongside DBLP, at the cost of admitting material that has not undergone peer review. These sources are at the same time computer science venues, five arXiv categories, CS publishers in DBLP, and agent-focused documentation, so design fields outside digital technology enter only where they publish in such venues. The domain boundary visible in Section 2.2, where the domains are digital subfields apart from robotics and control, mirrors this search surface in part, and the survey's claims about domains are correspondingly stronger inside digital technology than outside it.

The sources exclude work that none of them indexes, including non-English publications, and the query families define the vocabulary of the search, so work phrased outside that vocabulary can be missed. The field moves quickly, and the conclusions should be revisited as new evidence accumulates. The final corpus reflects the recency of the field: 14 of its 20 papers are 2026 preprints, three are from 2025, three from 2024, and none is earlier than 2024. A preprint can be revised or withdrawn, and a revision that changes a design's grounding claims would change the evidence behind Sections 4 and 5, though not the shape of the gap those sections describe. The corpus should therefore be revisited as the field's publication record settles, and the identifiers in the references are the versions retrieved during the search window.

6.6.2 Error in the abstract-level screens

The abstract-level screens carry classification error. The scope screen separated 2,987 included from 1,244 excluded records, and its excluded side was audited on a stratified sample of 59 records at 93.2% agreement. The quadrant screen reached an audit agreement of 85.7% on a stratified sample of the non-core quadrants, with an abstract-level miss rate of 6.1% as an upper bound. The three records misclassified by the abstract screen were all excluded at the full-text stage, so the final corpus is unaffected by that particular error. The rate itself is unstable, since it rests on three cases, and the extrapolated figure is larger than the observed loss: 6.1% of the non-core quadrants would amount to of the order of 150 candidates at the abstract level, none of which is shown to be a Design Grounding study. Two of the three reclassified records came from the Design-only stratum, which also has the lowest agreement (78.6%), so an abstract screen of this kind is most likely to miss a study that sits just inside the Design Grounding boundary. A re-run of the screens would move the final counts, and the miss rate above bounds how far; the three records recovered by the audit suggest the movement would be small enough to leave the qualitative conclusions, the dominance of G2 and the concentration of the four difficulties unchanged.

6.6.3 Extraction labels without human verification

The labels behind the two corpora of Section 2 were not independently verified. The application domain, the difficulty statement, and the evaluation strategy of the 184 Design-only and 192 Grounding-only papers were extracted by qwen3.7-max, and no systematic human spot-check of those labels was performed. The audit agreement of 85.7% reported in Section 3.1.2 is a cross-model check on a different input, since deepseek-v4-pro re-judged the same records from their full texts, and it therefore mixes a change of model with a change of evidence rather than measuring accuracy against human judgment. Two properties bound the risk without removing it: the assistant was restricted to statements that appear in the extracted text, and the difficulty statements behind Table 3 are reproduced in Online Resource 2, so each classification can be traced to its source. The difficulty labels carry two further limits. The pipeline records one stated difficulty per paper, and the papers in the corpus usually state more than one, so a different reading of the same paper records a different statement, and with it a different category. The extraction ran before the analysis, with the prompt reproduced in Appendix A, so the recorded statement was not chosen with the argument in view; the sixteen categories, however, overlap at their edges, and a limitation about what a design cannot be checked against can be read as a semantic-correctness complaint or as a lack of evaluation. Table 3 is read in that light, as the area the recorded difficulties occupy rather than as a partition of the papers.

6.6.4 Single-pass full-text adjudication

The full-text adjudication was not double-reviewed. deepseek-v4-pro, which was observed in testing to favor broad inclusion, judged the full texts, and no second model re-checked its inclusions. Low-confidence records were handled separately: a

cross-model re-scoring of 91 records promoted six, and the manual review excluded all six.

6.6.5 Single-judge grading and corpus size

The manual review is a single-judge process. The L/G grades were assigned by the lead researcher with no second human annotator for agreement, and the levels above the inclusion threshold record how far a paper reaches along each scale rather than separating one category from another. The final corpus of 20 papers is small, which bounds the strength of the mechanism analysis of Section 4.

7 Conclusion

The starting point of this survey is a shift in what agents do: from executing well-specified tasks to deciding what and how to build, which turns the agent into a producer of system specifications. A specification carries assumptions about the world it will meet, and those assumptions can be wrong, as the renovation blueprint of Section 1 is wrong about the walls being dry. Design Grounding names the problem of keeping the specification connected to the world it must satisfy. The design side of the problem appears across engineering, science, and computing, while the mechanisms that could address it have developed separately in software engineering, robotics, natural language processing, cognitive science, and other fields.

The evidence has a clear structure. A four-quadrant search over 2,987 records places Design Grounding as the smallest quadrant at 6.7%, and the 20 papers that survive full-text adjudication and manual review close the loop almost entirely through simulation feedback, with a single case reaching real measurement. The near misses show strong designs checked against worlds that are constructed, learned, or mediated by a human. Section 4 explains these observations through the structure of the design setting and the limits of the model, which make the loop hard to close, and through the cost of design-stage error, which makes the gap consequential. The directions of Section 5 follow the same order: differentiable and calibrated observation, institutional and detection-based authority, structural encodings and attribution, and process-level measurement. Their maturity differs, and the per-stage methods exist in adjacent fields, but no framework assembles them for the design setting. This assembly gap is the structural reason why the Design Grounding quadrant remains small, and it reframes the problem: the difficulty lies in integration rather than in missing methods.

The survey therefore concludes that Design Grounding is a cross-domain capability gap between what agents can generate and what they can validate, one that will likely close only when observation, validation, revision, and measurement are assembled into a single design process.

8 Outlook

The assembly gap of Section 5.5 suggests an agenda with three horizons.

- In the near term, the hypothesis can be tested by transplanting: the differentiable simulators, process supervision signals, and independent evaluation chains of adjacent fields map onto the four stages of the loop, and a design setting that adopts them can show which stages close first and where the transfer breaks.
- In the medium term, the bottleneck is institutional: a design-level counterpart of the foundry's validation authority, benchmarks that expose the Internal-to-External Bias rather than reward internal coherence, and process metrics with a standardized form.
- In the longer term, one can envision an agentic design process in which observation, validation, revision, and measurement run within a single closed loop, with the human positioned outside the loop as the auditor of the evaluation world.

Such a process would change what a future survey would find: the Design Grounding quadrant would grow, and the boundary between design and research automation would shift as validation becomes part of the design process itself. The renovation metaphor of Section 1 closes the survey where it began: the blueprint must return to the site, and in the agentic setting the site must be measured by instruments that the agent cannot rewrite.

Supplementary information. Supplementary information accompanies this article as data files rather than as a separate document. Online Resource 1 provides the extraction per-paper for both corpora of Section 2. Online Resource 2 provides the difficulty classification per-paper of the 184 Design-only papers, with the extracted statement, its assigned category, and the reason for the assignment. Online Resource 3 is a reproduction package that holds search records, screening and adjudication outputs, manual review records, and scripts behind every table.

Statements and Declarations

Funding: No funding was received for conducting this study.

Conflict of interest/Competing interests: The authors declare no competing interests.

Ethics approval and consent to participate: Not applicable.

Consent for publication: Not applicable.

Data availability: The datasets generated and analyzed during this study are provided as supplementary data. Online Resource 1 holds the per-paper extraction behind Tables 1 to 5; Online Resource 2 holds the per-paper difficulty classification behind Table 3; Online Resource 3 holds the search records, the four-quadrant screening results, the full-text adjudication records, and the manual review judgments behind Tables 6, 7, 9, and 10, with the corrections to three Design labels made in a second pass.

Materials availability: Not applicable.

Code availability: The scripts that build the corpora, run the screening, and produce the analysis behind the tables of this article are provided in Online Resource 3. The four-quadrant landscape of Figure 5 and the quadrant growth of Figure 6 are generated by two of them, and the remaining figures were drawn in external tools.

Declaration of AI use: LLM-based assistants performed the screening, extraction, and classification described in Section 3.1, subject to the manual review reported there. The illustrations of Figures 1 and 3, the wave band of Figure 2, and the board of Figure 9 were created with an AI image generation tool, as their captions state. The authors reviewed all outputs and take responsibility for the content of the manuscript.

Author contribution: Z.W. contributed to Conceptualization, Methodology, Software, Investigation, Data curation, Formal analysis, Visualization, and Writing (original draft). Y.L. contributed to Conceptualization, Supervision, and Writing (review and editing). Both authors read and approved the final manuscript.

Appendix A Screening Prompts

This appendix reproduces verbatim the four prompts used in the screening pipeline of Section 3, together with the two prompts used for the corpus extraction of Section 2 and the prompt used to classify the reported difficulties. The scope prompt drives the binary scope filter, the four-quadrant prompt drives the quadrant screening of Section 3.1.2, and the full-text eligibility prompt drives the adjudication of Section 3.1.3. The audit prompt is the full-text variant of the four-quadrant prompt, used for the stratified audit of the non-core quadrants. The medium- and low-confidence retest reuses the four-quadrant prompt unchanged. Each prompt includes its output schema, and all JSON responses were validated against the schema before being written to the result files.

A.1 Scope Screening Prompt

```
You are screening records for a literature review on LLM-based agentic
    systems.

Your only task is to decide whether the paper falls within the scope of
    LLM-based Agent research. Do not judge Design, Grounding, relevance to
    engineering, or eligibility for the final review corpus.

## Scope Definition

An LLM-based Agent is an agentic system in which a large language model,
    multimodal language model, or language-centered foundation model has a
    substantive role in goal-directed reasoning, decision-making, planning,
    tool use, environment interaction, autonomous adaptation, or coordination
    among agents.

A paper is within scope when it proposes, studies, evaluates, analyzes, or
    systematically reviews such an LLM-based agentic system.

## Decision Rule

Return include when the title or abstract supports the scope definition.
```

```
Return exclude when the supplied information clearly shows that the paper
    does not study an LLM-based agentic system.

When the supplied information is insufficient to confidently exclude the
    paper, return include to preserve recall for later screening.

Base the decision only on the supplied title and abstract. Do not infer an
    agentic role from unrelated uses of the word "agent". Do not use the
    search query provenance as evidence.

## Output

Return one JSON object and no additional text:

{
  "scope_decision": "<include|exclude>",
  "confidence": "<high|medium|low>",
  "scope_evidence": ["<short exact excerpt from the title or abstract>"],
  "reasoning": "<brief explanation of whether an LLM or language-centered
    foundation model has a substantive agentic role>"
}
```

A.2 Four-Quadrant Screening Prompt

```
You are screening literature for a review on grounding challenges in
    LLM-based agentic design.

Classify each paper by the relationship between Agentic Design and Grounding.
    Use only the supplied title and abstract. Do not infer mechanisms or
    causal relationships that are not stated or strongly supported by the
    text.

## Definitions

### Design

Design is the specification of an engineered system. It describes what is to
    be built through its architecture, components, interfaces, topology,
    constraints, geometry, or parameters.

For screening purposes, Design requires the agent to determine the
    constitutive structure of the engineered system. If the agent's primary
    role is to implement, repair, translate, or execute a system whose
    specification is externally determined, the Design condition is not met.
    When the abstract does not distinguish between these roles, mark the
    Design judgment as unclear.

### External World
```

```
The External World is the reality outside the agent's internal generative
    process. It has four properties:

1. It exists independently of the agent's design and evaluation framework.
    Its conditions, constraints, and consequences are not constructed for the
    purpose of measuring or testing the agent.
2. It lies outside the agent's internal semantic reasoning.
3. It cannot be freely rewritten by the agent.
4. It can expose incorrect or unrealistic internal judgments through
    observable referents, constraints, or consequences.

An External World referent is encountered, observed, or discovered. It is not
    designed or fabricated for the purpose of evaluation.

### Grounding

Grounding establishes correspondence between internal representations and
    independently existing External World referents through observations,
    interactions, or externally anchored feedback. A paper may study
    Grounding theoretically or implement it as a calibration mechanism.

### Design Grounding

Design Grounding occurs when feedback anchored in the External World
    evaluates an agent-generated Design and is used by the agentic process to
    revise, select, reject, or terminate that same Design.

The presence of Design and Grounding in the same paper is insufficient. The
    Grounding feedback must directly target the agent-generated Design.

## Judgment Values

Use the following values for each judgment:

- yes: the title or abstract provides clear supporting evidence;
- no: the title or abstract provides clear evidence that the condition is
    absent;
- unclear: the supplied information is insufficient to determine the
    condition without inference.

## Decision Process

1. Identify the primary artifact, representation, decision, or behavior
    generated or modified by the agent.
2. Determine whether the primary target satisfies the Design definition.
3. Determine whether the paper studies or implements Grounding.
4. Identify the direct target of Grounding.
5. Determine whether an External World reference is present.
6. Determine whether feedback from that reference is returned to the agentic
    design process and updates the same Design.
```

```
7. Assign a provisional quadrant.

## Provisional Quadrants

### design_grounding

The agent generates a Design and External World feedback directly updates,
    selects, rejects, or terminates that Design.

### design_only

The agent generates a Design and the Design itself does not enter a Grounding
    feedback loop, or the feedback relationship is unclear.

### grounding_only

The paper studies or implements Grounding while the grounded target is not an
    agent-generated Design.

### neither

The paper contains neither Agentic Design nor Grounding under the supplied
    definitions.

## Output

Return one JSON object and no additional text:

{
  "primary_artifact": "<the main artifact, representation, decision, or
    behavior>",
  "design_status": "<yes|no|unclear>",
  "grounding_status": "<yes|no|unclear>",
  "grounding_target": "<design|text_or_fact|
perception|action|task_plan|
state_or_world_model|implementation_code|
other|none|unclear>",
  "external_reference_status": "<yes|no|unclear>",
  "feedback_updates_design_status": "<yes|no|unclear>",
  "provisional_quadrant":
    "<design_grounding|design_only|grounding_only|neither>",
  "confidence": "<high|medium|low>",
  "design_evidence": ["<short exact evidence excerpt>"],
  "grounding_evidence": ["<short exact evidence excerpt>"],
  "feedback_coupling_evidence": ["<short exact evidence excerpt>"],
  "reasoning": "<brief explanation of the artifact, Grounding target,
    feedback relationship, and provisional quadrant>"
}
```

A.3 Full-Text Eligibility Prompt

```
You are performing full-text eligibility adjudication for a review on
    grounding challenges in LLM-based agentic design.

Read the supplied full text and extract evidence for each criterion. Base
    every judgment on explicit statements in the text. When the text is
    ambiguous or silent, mark the judgment as unclear rather than inferring.

## Core Definitions

### Design

Design is the specification of an engineered system. It describes what is to
    be built through its architecture, components, interfaces, topology,
    constraints, geometry, or parameters.

For eligibility purposes, Design requires the agent to determine the
    constitutive structure of the engineered system. If the agent's primary
    role is to implement, repair, translate, or execute a system whose
    specification is externally determined, the Design condition is not met.

### External World

The External World is the reality outside the agent's internal generative
    process. It has four properties:

1. It exists independently of the agent's design and evaluation framework.
    Its conditions, constraints, and consequences are not constructed for the
    purpose of measuring or testing the agent.
2. It lies outside the agent's internal semantic reasoning.
3. It cannot be freely rewritten by the agent.
4. It can expose incorrect or unrealistic internal judgments through
    observable referents, constraints, or consequences.

An External World referent is encountered, observed, or discovered. It is not
    designed or fabricated for the purpose of evaluation.

### Grounding

Grounding establishes correspondence between internal representations and
    independently existing External World referents through observations,
    interactions, or externally anchored feedback.

### Design Grounding

Design Grounding occurs when feedback anchored in the External World
    evaluates an agent-generated Design and is used by the agentic process to
    revise, select, reject, or terminate that same Design.
```

```
The presence of Design and Grounding in the same paper is insufficient. The
    Grounding feedback must directly target the agent-generated Design.

## Eligibility Criteria

A paper is eligible when ALL of the following are supported by the full text:

1. The paper studies an LLM-based agentic system.
2. The agent generates a Design (specification of an engineered system).
3. An External World reference is present and is used to evaluate the Design.
4. Feedback from that External World reference returns to the agentic process
    and is used to revise, select, reject, or terminate that same Design.

## Decision

- include: the full text provides clear evidence for all four criteria.
- exclude: the full text provides clear evidence that at least one criterion
    is not met.
- unclear: the full text is ambiguous or silent on one or more criteria, and
    the decision cannot be made without inference.

## Output

Return one JSON object and no additional text:

{
  "llm_agent_evidence": {
    "holds": true,
    "role": "<the agent's substantive role in the system>",
    "quote": "<exact excerpt from the full text>",
    "section": "<section or paragraph where the evidence was found>"
  },
  "design_evidence": {
    "holds": true,
    "artifact": "<the Design artifact generated by the agent>",
    "constitutive_decisions": ["<what the agent decided about the system's
    structure>"],
    "quote": "<exact excerpt>",
    "section": "<section or paragraph>"
  },
  "external_reference_evidence": {
    "holds": true,
    "source": "<the External World referent>",
    "independence": "<why this referent is not constructed for evaluation>",
    "quote": "<exact excerpt>",
    "section": "<section or paragraph>"
  },
  "feedback_coupling_evidence": {
    "holds": true,
    "receiver": "<who or what receives the feedback>",
```

```
    "design_change": "<how the Design was revised, selected, rejected, or
    terminated>",
    "quote": "<exact excerpt>",
    "section": "<section or paragraph>"
  },
  "eligibility": "<include|exclude|unclear>",
  "exclusion_reason": "<if exclude, which criterion failed and why>",
  "confidence": "<high|medium|low>"
}
```

A.4 Non-Core Quadrant Audit Prompt

```
You are auditing a previous literature screening for a review on grounding
    challenges in LLM-based agentic design. Each paper was previously
    classified using only its title and abstract. You now have access to the
    full text and may reach a different conclusion.

Classify each paper by the relationship between Agentic Design and Grounding.
    Use the supplied title and full text. Do not infer mechanisms or causal
    relationships that are not stated or strongly supported by the text. Your
    classification may differ from the original abstract-only classification.

## Definitions

### Design

Design is the specification of an engineered system. It describes what is to
    be built through its architecture, components, interfaces, topology,
    constraints, geometry, or parameters.

For screening purposes, Design requires the agent to determine the
    constitutive structure of the engineered system. If the agent's primary
    role is to implement, repair, translate, or execute a system whose
    specification is externally determined, the Design condition is not met.
    When the full text does not distinguish between these roles, mark the
    Design judgment as unclear.

### External World

The External World is the reality outside the agent's internal generative
    process. It has four properties:

1. It exists independently of the agent's design and evaluation framework.
    Its conditions, constraints, and consequences are not constructed for the
    purpose of measuring or testing the agent.
2. It lies outside the agent's internal semantic reasoning.
3. It cannot be freely rewritten by the agent.
4. It can expose incorrect or unrealistic internal judgments through
    observable referents, constraints, or consequences.
```

```
An External World referent is encountered, observed, or discovered. It is not
    designed or fabricated for the purpose of evaluation.

### Grounding

Grounding establishes correspondence between internal representations and
    independently existing External World referents through observations,
    interactions, or externally anchored feedback. A paper may study
    Grounding theoretically or implement it as a calibration mechanism.

### Design Grounding

Design Grounding occurs when feedback anchored in the External World
    evaluates an agent-generated Design and is used by the agentic process to
    revise, select, reject, or terminate that same Design.

The presence of Design and Grounding in the same paper is insufficient. The
    Grounding feedback must directly target the agent-generated Design.

## Judgment Values

Use the following values for each judgment:

- yes: the full text provides clear supporting evidence;
- no: the full text provides clear evidence that the condition is absent;
- unclear: the supplied information is insufficient to determine the
    condition without inference.

## Decision Process

1. Identify the primary artifact, representation, decision, or behavior
    generated or modified by the agent.
2. Determine whether the primary target satisfies the Design definition.
3. Determine whether the paper studies or implements Grounding.
4. Identify the direct target of Grounding.
5. Determine whether an External World reference is present.
6. Determine whether feedback from that reference is returned to the agentic
    design process and updates the same Design.
7. Assign a provisional quadrant.

## Provisional Quadrants

### design_grounding

The agent generates a Design and External World feedback directly updates,
    selects, rejects, or terminates that Design.

### design_only
```

```
The agent generates a Design and the Design itself does not enter a Grounding
    feedback loop, or the feedback relationship is unclear.

### grounding_only

The paper studies or implements Grounding while the grounded target is not an
    agent-generated Design.

### neither

The paper contains neither Agentic Design nor Grounding under the supplied
    definitions.

## Output

Return one JSON object and no additional text:

{
  "primary_artifact": "<the main artifact, representation, decision, or
    behavior>",
  "design_status": "<yes|no|unclear>",
  "grounding_status": "<yes|no|unclear>",
  "grounding_target": "<design|text_or_fact|
perception|action|task_plan|
state_or_world_model|implementation_code|
other|none|unclear>",
  "external_reference_status": "<yes|no|unclear>",
  "feedback_updates_design_status": "<yes|no|unclear>",
  "provisional_quadrant":
    "<design_grounding|design_only|grounding_only|neither>",
  "confidence": "<high|medium|low>",
  "design_evidence": ["<short exact evidence excerpt>"],
  "grounding_evidence": ["<short exact evidence excerpt>"],
  "feedback_coupling_evidence": ["<short exact evidence excerpt>"],
  "reasoning": "<brief explanation of the artifact, Grounding target,
    feedback relationship, and provisional quadrant>"
}
```

A.5 Design-Only Extraction Prompt

```
You are analyzing a collection of papers about LLM-based Agentic Design. Each
    paper describes an LLM agent that designs or specifies an engineered
    system.

For each paper, extract the following information from the supplied title,
    abstract, and selected sections. Do NOT infer information not present in
    the text. Just report what the paper actually says.

## Domain Classification
```

```
Assign the paper to ONE of the following domains based on the engineered
    system being designed:

- hardware_eda: Chip design, FPGA, HLS, circuits, accelerators, electronic
    design automation
- software_engineering: Software architecture, code generation, testing,
    refactoring, API design
- scientific_computing: PDE solvers, molecular dynamics, materials, quantum
    computing, numerical methods
- network_telecom: 6G, RAN, network protocols, communication systems
- robotics_control: Robots, control systems, autonomous vehicles, motion
    planning
- agent_architecture: Agent frameworks, multi-agent coordination, agent tool
    design, agent orchestration
- ai_model_design: Neural architecture search, model design, training
    pipeline design
- other_engineering: Mechanical, civil, chemical, aerospace, or other
    engineering design
- other: Does not fit the above categories

## Difficulty and Gap Extraction

From the Limitations, Discussion, or Conclusion sections, extract:

- artifact_type: What the agent designs (e.g., "hardware accelerator
    architecture", "multi-agent coordination protocol"). One sentence.
- core_difficulty: The main difficulty, limitation, or challenge the paper
    acknowledges. This should come from the sections where authors discuss
    what did not work well, what remains unsolved, or what future work is
    needed. One sentence.
- evaluation_gap: How the paper evaluates its design output. Describe the
    evaluation approach and what it does or does not validate. One sentence.
    If evaluation is not discussed, say "not discussed".

## Output

Return one JSON object and no additional text:

{
  "domain": "<one of the domain categories above>",
  "artifact_type": "<what the agent designs - one sentence>",
  "core_difficulty": "<the main difficulty, limitation, or challenge the
    paper acknowledges - one sentence extracted or paraphrased from the
    Limitations/Discussion/Conclusion>",
  "evaluation_gap": "<how the design output is evaluated - one sentence>",
  "confidence": "<high|medium|low>"
}
```

A.6 Grounding-Only Extraction Prompt

This prompt names none of the terms of this survey, so the grounding target labels it produces are not conditioned on the Design Grounding framing; design is one option among seven. Both extraction prompts receive the same inputs: the title, the abstract, and the part of the paper that reports limitations, discussion, or conclusions, located by heading, or the closing part of the text when no such heading is found.

```
You are analyzing a collection of research papers about grounding in AI
    systems. Each paper works on establishing or maintaining a correspondence
    between an AI system's internal representations and some external reality
    through observations, interactions, or external feedback.

For each paper, extract the following information from the supplied title,
    abstract, and selected sections. Do NOT infer information not present in
    the text. Report what the paper actually says.

## Application Domain

Assign the paper to ONE of the following domains based on its application
    context:

- visual_perception: Visual grounding, VLM perception, multimodal
    understanding, image/video grounding
- embodied_agent: Vision-language navigation (VLN), robotics, embodied
    agents, manipulation, autonomous driving
- nlp_factuality: Factuality, retrieval-augmented generation, knowledge
    grounding, hallucination mitigation in text
- agent_tool_execution: Tool use, API calling, code execution, environment
    interaction by agents
- state_estimation: SLAM, world models, state/pose estimation, belief tracking
- domain_science: Vertical applications (medical, materials, bio, chemistry,
    finance, weather, etc.)
- dialogue_instruction: Dialogue, instruction following, preference/user
    intent
- game_simulation: Games, simulation environments, benchmarks
- other: Does not fit the above categories

## Grounding Target

Identify what the system's representations are being calibrated against, or
    what is being anchored. Assign ONE of the following categories:

- perception: Visual, audio, or multimodal perception of the environment
- text_or_fact: Text, factual claims, knowledge, source documents
- action: Physical or virtual actions, behaviors, control outputs
- task_plan: High-level task plans, sequences of steps, long-horizon goals
- state_or_world_model: World state, pose, environment model, belief state
- design: A system, architecture, product, or interface design/specification
    produced by the system
```

```
- other: Anything not fitting the above

## Difficulty (one sentence)

From the Limitations, Discussion, or Conclusion sections, summarize in ONE
    sentence the main difficulty, limitation, or challenge the paper faces in
    achieving grounding.

## Output

Return one JSON object and no additional text:
{
  "application_domain": "<one of the domain categories>",
  "grounding_target": "<one of the grounding target categories>",
  "mechanism": "<the grounding mechanism used: e.g. retrieval, affordance,
    sensor feedback, sim-to-real, tool execution, self-verification, human
    annotation, contrastive training... one short phrase>",
  "external_source": "<what serves as the external reference: e.g. knowledge
    base, simulator, physical sensors, real environment, human labels... one
    short phrase>",
  "core_difficulty": "<main difficulty in achieving grounding, one sentence
    from Limitations/Discussion/Conclusion>",
  "representative": "<one-sentence takeaway of the paper's grounding
    approach>",
  "confidence": "<high|medium|low>"
}
```

A.7 Difficulty Classification Prompt

The category set below is set out in advance, and the assistant assigns a category and a one-line reason to each difficulty statement extracted by the Design-only extraction prompt.

```
You classify the reported difficulty of an LLM-agentic-design paper into one
    category.

The paper produces a design (architecture, topology, formulation, workflow,
    etc.) with an LLM agent.
The difficulty is a one-sentence statement extracted from the paper's
    Limitations/Discussion/Conclusion sections.

Categories and definitions:
- scalability: Designs that work on small instances fail as the problem or
    system grows; combinatorial, resource, or context-window limits
- structural_change: Revising a design is costly once fixed; changes
    propagate through components, interfaces, or topology
- expert_dependency: Human designers supply domain knowledge, constraints, or
    validation that the environment should provide
- generalization: Designs fail when carried to settings different from the
    evaluation setting; out-of-distribution, transfer, robustness
```

```
- sim_to_real_gap: Internal evaluation (simulation, benchmark) diverges from
    external behavior; deployment, field conditions
- hallucination: Agent produces internally coherent but externally infeasible
    or fabricated outputs
- feedback_quality: Available feedback is partial, delayed, noisy, or
    circular; self-evaluation limitations
- long_horizon: Consequences of design decisions unfold over long time
    scales; multi-step error propagation
- data_scarcity: Insufficient or low-quality domain data for the design task
- lack_or_limited_evaluation: Evaluation is absent, limited, or proxied; hard
    to evaluate the design
- domain_knowledge: Agent lacks domain-specific knowledge or
    physical/engineering constraints needed for the design
- semantic_correctness: Design output lacks verifiable semantic correctness,
    physical consistency, or feasibility
- reasoning_limitation: Agent reasoning is shallow, over-anchored, or lacks
    global/system-level foresight
- cost_or_efficiency: Computational cost, latency, inference time, or
    resource overhead limit the design process
- no_difficulty_extracted: The paper does not state a difficulty explicitly
    (no Limitations/Discussion section)
- other: None of the above

Rules:
- Assign exactly ONE category. Choose the best fit based on the difficulty
    statement itself.
- If the statement matches no category, use "other".
- If the paper does not state a difficulty, use "no_difficulty_extracted".
- Reason: one short clause (under 15 words) explaining the assignment.

Return JSON: {"category": "<one of the categories>", "reason": "<short
    reason>"}
```

Appendix B Search Queries

The 128 queries are generated from four families of term combinations: 48 Grounding × Agent queries, 56 Synonym × Agent queries, 4 Grounding-only queries, and 20 Design-centric queries, covering the period 2020 to 2026. The four families and their roles are summarized in Table 6; the complete list follows. Each entry is a term combination rather than a literal search string: the two terms are submitted to the arXiv and DBLP interfaces as a conjunction, and the scripts that translate these combinations into API requests accompany this article.

B.1 Grounding x Agent (48)

No.	Query
1	grounding_x_agent: symbol grounding ; agentic design
2	grounding_x_agent: symbol grounding ; LLM-based agent
3	grounding_x_agent: symbol grounding ; LLM agent
4	grounding_x_agent: symbol grounding ; agent architecture
5	grounding_x_agent: symbol grounding ; coding agent
6	grounding_x_agent: symbol grounding ; autonomous design agent
7	grounding_x_agent: symbol grounding ; design agent
8	grounding_x_agent: symbol grounding ; software design agent
9	grounding_x_agent: language grounding ; agentic design
10	grounding_x_agent: language grounding ; LLM-based agent
11	grounding_x_agent: language grounding ; LLM agent
12	grounding_x_agent: language grounding ; agent architecture
13	grounding_x_agent: language grounding ; coding agent
14	grounding_x_agent: language grounding ; autonomous design agent
15	grounding_x_agent: language grounding ; design agent
16	grounding_x_agent: language grounding ; software design agent
17	grounding_x_agent: affordance grounding ; agentic design
18	grounding_x_agent: affordance grounding ; LLM-based agent
19	grounding_x_agent: affordance grounding ; LLM agent
20	grounding_x_agent: affordance grounding ; agent architecture
21	grounding_x_agent: affordance grounding ; coding agent
22	grounding_x_agent: affordance grounding ; autonomous design agent
23	grounding_x_agent: affordance grounding ; design agent
24	grounding_x_agent: affordance grounding ; software design agent
25	grounding_x_agent: visual grounding ; agentic design
26	grounding_x_agent: visual grounding ; LLM-based agent
27	grounding_x_agent: visual grounding ; LLM agent
28	grounding_x_agent: visual grounding ; agent architecture
29	grounding_x_agent: visual grounding ; coding agent
30	grounding_x_agent: visual grounding ; autonomous design agent
31	grounding_x_agent: visual grounding ; design agent
32	grounding_x_agent: visual grounding ; software design agent
33	grounding_x_agent: grounding ; agentic design
34	grounding_x_agent: grounding ; LLM-based agent
35	grounding_x_agent: grounding ; LLM agent
36	grounding_x_agent: grounding ; agent architecture
37	grounding_x_agent: grounding ; coding agent
38	grounding_x_agent: grounding ; autonomous design agent
39	grounding_x_agent: grounding ; design agent
40	grounding_x_agent: grounding ; software design agent
41	grounding_x_agent: grounded ; agentic design
42	grounding_x_agent: grounded ; LLM-based agent
43	grounding_x_agent: grounded ; LLM agent
44	grounding_x_agent: grounded ; agent architecture
45	grounding_x_agent: grounded ; coding agent
46	grounding_x_agent: grounded ; autonomous design agent
47	grounding_x_agent: grounded ; design agent
48	grounding_x_agent: grounded ; software design agent

B.2 Synonym x Agent (56)

No.	Query
1	synonym_x_agent: hallucination ; agentic design
2	synonym_x_agent: hallucination ; LLM-based agent
3	synonym_x_agent: hallucination ; LLM agent

Continued

No.	Query
4	`synonym_x_agent: hallucination ; agent architecture`
5	`synonym_x_agent: hallucination ; coding agent`
6	`synonym_x_agent: hallucination ; autonomous design agent`
7	`synonym_x_agent: hallucination ; design agent`
8	`synonym_x_agent: hallucination ; software design agent`
9	`synonym_x_agent: calibration ; agentic design`
10	`synonym_x_agent: calibration ; LLM-based agent`
11	`synonym_x_agent: calibration ; LLM agent`
12	`synonym_x_agent: calibration ; agent architecture`
13	`synonym_x_agent: calibration ; coding agent`
14	`synonym_x_agent: calibration ; autonomous design agent`
15	`synonym_x_agent: calibration ; design agent`
16	`synonym_x_agent: calibration ; software design agent`
17	`synonym_x_agent: constraint ; agentic design`
18	`synonym_x_agent: constraint ; LLM-based agent`
19	`synonym_x_agent: constraint ; LLM agent`
20	`synonym_x_agent: constraint ; agent architecture`
21	`synonym_x_agent: constraint ; coding agent`
22	`synonym_x_agent: constraint ; autonomous design agent`
23	`synonym_x_agent: constraint ; design agent`
24	`synonym_x_agent: constraint ; software design agent`
25	`synonym_x_agent: verification ; agentic design`
26	`synonym_x_agent: verification ; LLM-based agent`
27	`synonym_x_agent: verification ; LLM agent`
28	`synonym_x_agent: verification ; agent architecture`
29	`synonym_x_agent: verification ; coding agent`
30	`synonym_x_agent: verification ; autonomous design agent`
31	`synonym_x_agent: verification ; design agent`
32	`synonym_x_agent: verification ; software design agent`
33	`synonym_x_agent: specification gaming ; agentic design`
34	`synonym_x_agent: specification gaming ; LLM-based agent`
35	`synonym_x_agent: specification gaming ; LLM agent`
36	`synonym_x_agent: specification gaming ; agent architecture`
37	`synonym_x_agent: specification gaming ; coding agent`
38	`synonym_x_agent: specification gaming ; autonomous design agent`
39	`synonym_x_agent: specification gaming ; design agent`
40	`synonym_x_agent: specification gaming ; software design agent`
41	`synonym_x_agent: faithfulness ; agentic design`
42	`synonym_x_agent: faithfulness ; LLM-based agent`
43	`synonym_x_agent: faithfulness ; LLM agent`
44	`synonym_x_agent: faithfulness ; agent architecture`
45	`synonym_x_agent: faithfulness ; coding agent`
46	`synonym_x_agent: faithfulness ; autonomous design agent`
47	`synonym_x_agent: faithfulness ; design agent`
48	`synonym_x_agent: faithfulness ; software design agent`
49	`synonym_x_agent: factuality ; agentic design`
50	`synonym_x_agent: factuality ; LLM-based agent`
51	`synonym_x_agent: factuality ; LLM agent`
52	`synonym_x_agent: factuality ; agent architecture`
53	`synonym_x_agent: factuality ; coding agent`
54	`synonym_x_agent: factuality ; autonomous design agent`
55	`synonym_x_agent: factuality ; design agent`
56	`synonym_x_agent: factuality ; software design agent`

B.3 Grounding-only (4)

No.	Query
1	grounding_only: symbol grounding
2	grounding_only: language grounding
3	grounding_only: affordance grounding
4	grounding_only: visual grounding

B.4 Design-centric (20)

No.	Query
1	design_centric: agentic design
2	design_centric: LLM-based design agent
3	design_centric: LLM design agent
4	design_centric: autonomous design agent
5	design_centric: LLM ; engineering design
6	design_centric: LLM ; inverse design
7	design_centric: LLM ; design space exploration
8	design_centric: LLM ; generative design
9	design_centric: LLM ; co-design
10	design_centric: LLM ; simulation-driven design
11	design_centric: LLM ; automated engineering design
12	design_centric: LLM agent ; system design
13	design_centric: LLM agent ; software architecture
14	design_centric: LLM agent ; hardware design
15	design_centric: LLM agent ; circuit design
16	design_centric: LLM agent ; CAD
17	design_centric: LLM agent ; robot design
18	design_centric: LLM agent ; material design
19	design_centric: LLM agent ; AI system design
20	design_centric: LLM agent ; agent architecture

References

Abou Ali M, Dornaika F, Charafeddine J (2025) Agentic AI: a comprehensive survey of architectures, applications, and future directions. Artificial Intelligence Review 59(1):11. https://doi.org/10.1007/s10462-025-11422-4

Ahn M, Brohan A, Brown N, et al (2022) Do as I can, not as I say: Grounding language in robotic affordances. URL https://arxiv.org/abs/2204.01691, arXiv preprint arXiv:2204.01691

Allam A, Mansour Y, Shalan M (2025) ASIC-agent: An autonomous multi-agent system for ASIC design with benchmark evaluation. URL https://arxiv.org/abs/2508.15940, arXiv preprint arXiv:2508.15940 [cs.AR]

Bao Z, Lin Z, Wang J, et al (2026) AnalogAgent: Self-improving analog circuit design automation with LLM agents. URL https://arxiv.org/abs/2603.23910, arXiv preprint arXiv:2603.23910 [cs.AR]

Berger E, Usama M, Mehlstäubl J, et al (2026) Physics-in-the-loop: A hybrid agentic architecture for validated CAD engineering design. URL https://arxiv.org/abs/2605.19717, arXiv preprint arXiv:2605.19717 [cs.CE]

Bhandwaldar A, Choudhury M, Puri R, et al (2026) Agent factories for high level synthesis: How far can general-purpose coding agents go in hardware optimization? URL https://arxiv.org/abs/2603.25719, arXiv preprint arXiv:2603.25719 [cs.AR]

Bonagiri A, Borkar D, Anderias GJ, et al (2026) CausalFlow: Causal attribution and counterfactual repair for LLM agent failures. URL https://arxiv.org/abs/2605.25338, arXiv preprint arXiv:2605.25338

Bondarenko Y, et al (2025) Palisade: Measuring specification gaming in reasoning models. URL https://arxiv.org/abs/2503.19000, arXiv preprint arXiv:2503.19000

Cadena C, Carlone L, Carrillo H, et al (2016) Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics 32(6):1309–1332. https://doi.org/10.1109/TRO.2016.2624754

Chen S, Shu T, Ren S (2026) Learning design skills as memory policies for agentic photonic inverse design. URL https://arxiv.org/abs/2605.29421, arXiv preprint arXiv:2605.29421 [physics.optics]

Coelho Mollo D, Millière R (2026) The vector grounding problem. Philosophy and the Mind Sciences 7(1). https://doi.org/10.33735/phimisci.2026.12307

Cohen V, Liu JX, Mooney R, et al (2024) A survey of robotic language grounding: Tradeoffs between symbols and embeddings. In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI), pp 7999–8009, https://doi.org/10.24963/ijcai.2024/885

Collins J, Howard D, Leitner J (2020) Quantifying the reality gap in robotic manipulation tasks. URL https://arxiv.org/abs/1811.01484, arXiv preprint arXiv:1811.01484

Crilly AJ, Travis P, Brodrick JP, et al (2026) High-dimensional inverse design of inertial fusion implosions via differentiable simulation. URL https://arxiv.org/abs/2606.08827, arXiv preprint arXiv:2606.08827

Da Ros F, Soprano M, Di Gaspero L, et al (2025) Large language models for combinatorial optimization: A systematic review. URL https://arxiv.org/abs/2507.03637, arXiv preprint arXiv:2507.03637

Dijkstra EW (1982) On the role of scientific thought. In: Selected Writings on Computing: A Personal Perspective. Springer, p 60–66, https://doi.org/10.1007/978-1-4612-5695-3_12

Do B, Zhang R (2023) Multi-fidelity Bayesian optimization: A review. URL https://arxiv.org/abs/2311.13050, arXiv preprint arXiv:2311.13050

Fan A, Gokkaya B, Harman M, et al (2023) Large language models for software engineering: Survey and open problems. In: Proceedings of the IEEE/ACM International Conference on Software Engineering: Future of Software Engineering. IEEE, pp 31–53, https://doi.org/10.1109/ICSE-FoSE59343.2023.00008

Gridach M, Nanavati J, Zine El Abidine K, et al (2025) Agentic AI for scientific discovery: A survey of progress, challenges, and future directions. URL https://arxiv.org/abs/2503.08979, arXiv preprint arXiv:2503.08979

Harnad S (1990) The symbol grounding problem. Physica D: Nonlinear Phenomena 42(1):335–346. https://doi.org/10.1016/0167-2789(90)90087-6, URL https://www.sciencedirect.com/science/article/pii/0167278990900876

Hong S, Zhuge M, Chen J, et al (2024) MetaGPT: Meta programming for a multi-agent collaborative framework. In: International Conference on Learning Representations (ICLR), arXiv preprint arXiv:2308.00352

Hou X, Zhao Y, Liu Y, et al (2024) Large language models for software engineering: A systematic literature review. ACM Trans Softw Eng Methodol 33(8):1–79. https://doi.org/10.1145/3695988

Huang L, Yu W, Ma W, et al (2025) A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. URL https://arxiv.org/abs/2311.05232, arXiv preprint arXiv:2311.05232

Huotala A, Kuutila M, Ralph P, et al (2024) The promise and challenges of using LLMs to accelerate the screening process of systematic reviews. URL https://arxiv.org/abs/2404.15667, arXiv preprint arXiv:2404.15667

INCOSE (2023) Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, 5th edn. Wiley, Hoboken, NJ

International Organization for Standardization (2015) Systems and software engineering — system life cycle processes. URL https://www.iso.org/standard/63711.html, ISO/IEC/IEEE 15288:2015

Ji Z, Lee N, Frieske R, et al (2023) Survey of hallucination in natural language generation. ACM Computing Surveys 55(12):1–38. https://doi.org/10.1145/3571730

Jiang S, Xie M, Chen FY, et al (2025) Intelligent design 4.0: Paradigm evolution toward the agentic AI era. https://doi.org/10.48550/arXiv.2506.09755, URL https://arxiv.org/abs/2506.09755, arXiv preprint arXiv:2506.09755 [cs.CE]

Kaelbling LP, Littman ML, Cassandra AR (1998) Planning and acting in partially observable stochastic domains. Artificial Intelligence 101(1-2):99–134. https://doi.org/10.1016/S0004-3702(98)00023-X

Kenthapadi K, Sameki M, Taly A (2024) Grounding and evaluation for large language models: Practical challenges and lessons learned (survey). URL https://arxiv.org/abs/2407.12858, arXiv preprint arXiv:2407.12858

Khraisha Q, Put S, Kappenberg J, et al (2024) Can large language models replace humans in the systematic review process? evaluating GPT-4's efficacy in screening and extracting data from peer-reviewed and grey literature in multiple languages. URL https://arxiv.org/abs/2310.17526, arXiv preprint arXiv:2310.17526

Krakovna V, Uesato J, Mikulik V, et al (2020) Specification gaming: the flip side of AI ingenuity. URL https://deepmind.google/discover/blog/specification-gaming-the-flip-side-of-ai-ingenuity/, DeepMind Blog

Krishna R, Krishna S, Chin D (2026) Design conductor: An agent autonomously builds a 1.5 GHz linux-capable RISC-V CPU. URL https://arxiv.org/abs/2603.08716, arXiv preprint arXiv:2603.08716 [cs.AR]

Lai Y, Lee S, Chen G, et al (2024) AnalogCoder: Analog circuit design via training-free code generation. URL https://arxiv.org/abs/2405.14918, arXiv preprint arXiv:2405.14918 [cs.AR]

Lee EA (2008) Cyber physical systems: Design challenges. In: Proceedings of the 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC). IEEE, pp 363–369, https://doi.org/10.1109/ISORC.2008.25

Lewis P, Perez E, Piktus A, et al (2020) Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems, pp 9459–9474

Li G, Hammoud HAAK, Itani H, et al (2023) CAMEL: Communicative agents for "mind" exploration of large language model society. In: Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:2303.17760

Lie KA, Møyner O, Svee E, et al (2026) Agentic scientific simulation: Execution-grounded model construction and reconstruction. URL https://arxiv.org/abs/2603.00214, arXiv preprint arXiv:2603.00214 [physics.comp-ph]

Liu F, Yao Y, Guo P, et al (2026a) A systematic survey on large language models for algorithm design. ACM Computing Surveys 58(8):1–32. https://doi.org/10.1145/3787585

Liu S, Krentsel A, Agarwal S, et al (2026b) The time is here for just-in-time systems: Challenges and opportunities. URL https://arxiv.org/abs/2605.24096, arXiv preprint arXiv:2605.24096 [cs.OS]

Liu Z (2026) A unified formal theory on the logical limits of symbol grounding. Minds and Machines 36(1):11. https://doi.org/10.1007/s11023-026-09763-2

Lou Y, Lai L, Li S, et al (2026) SpecDB: LLM-generated customized databases via feature-oriented decomposition. URL https://arxiv.org/abs/2605.31097, arXiv preprint arXiv:2605.31097 [cs.DB]

Luo L, Liu Y, Liu R, et al (2024) Improve mathematical reasoning in language models by automated process supervision. URL https://arxiv.org/abs/2406.06592, arXiv preprint arXiv:2406.06592

Minsky M (1961) Steps toward artificial intelligence. Proceedings of the IRE 49(1):8–30. https://doi.org/10.1109/JRPROC.1961.287775

Nallathambi A, Knight C, Ganguly S, et al (2026) A3D: Agentic AI flow for autonomous accelerator design. URL https://arxiv.org/abs/2605.15237, arXiv preprint arXiv:2605.15237 [cs.AR]

Nath V, Raja P, Yoon C, et al (2025) ToolComp: A multi-tool reasoning & process supervision benchmark. URL https://arxiv.org/abs/2501.01290, arXiv preprint arXiv:2501.01290

Neider D, Roy R (2025) What is formal verification without specifications? a survey on mining LTL specifications. URL https://arxiv.org/abs/2501.16274, arXiv preprint arXiv:2501.16274

Pan A, Bhatia K, Steinhardt J (2022) The effects of reward misspecification: Mapping and mitigating misaligned models. URL https://arxiv.org/abs/2201.03544, arXiv preprint arXiv:2201.03544

Pan J, Zhou G, Chang CC, et al (2025) A survey of research in large language models for electronic design automation. ACM Transactions on Design Automation of Electronic Systems 30(3):1–21. https://doi.org/10.1145/3715324

Panickssery A, Bowman SR, Feng S (2024) LLM evaluators recognize and favor their own generations. URL https://arxiv.org/abs/2404.13076, arXiv preprint arXiv:2404.13076

Park JS, O'Brien JC, Cai CJ, et al (2023) Generative agents: Interactive simulacra of human behavior. In: Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST), arXiv preprint arXiv:2304.03442

Patra A, Rout S, Ravindran A (2024) AiEDA: Agentic AI design framework for digital ASIC system design. URL https://arxiv.org/abs/2412.09745, arXiv preprint arXiv:2412.09745 [cs.AR]

Pepe A, Lin CY, Magka D, et al (2026) Agentic discovery of neural architectures: AIRA-compose and AIRA-design. URL https://arxiv.org/abs/2605.15871, arXiv preprint arXiv:2605.15871 [cs.LG]

Pignatelli E, Ferret J, Geist M, et al (2023) A survey of temporal credit assignment in deep reinforcement learning. URL https://arxiv.org/abs/2312.01072, arXiv preprint arXiv:2312.01072

Rahman S, Rahman MR (2026) AgentPLM: Agentic protein language models with reasoning-augmented decoding for protein sequence design. URL https://arxiv.org/abs/2606.02386, arXiv preprint arXiv:2606.02386 [q-bio.BM]

Ren J, Li H, Liu Y, et al (2026) A blueprint for self-evolving coding agents in vehicle aerodynamic drag prediction. URL https://arxiv.org/abs/2603.21698, arXiv preprint arXiv:2603.21698 [cs.SE]

Romera-Paredes B, Barekatain M, Novikov A, et al (2024) Mathematical discoveries from program search with large language models. Nature 625(7995):468–475. https://doi.org/10.1038/s41586-023-06924-6

Schick T, Dwivedi-Yu J, Dessì R, et al (2023) Toolformer: Language models can teach themselves to use tools. URL https://arxiv.org/abs/2302.04761, arXiv preprint arXiv:2302.04761 [cs.CL]

Shailendra S, Kadel R, Sharma A, et al (2026) L-PRISMA: An extension of PRISMA in the era of generative artificial intelligence (GenAI). URL https://arxiv.org/abs/2603.19236, arXiv preprint arXiv:2603.19236

Shinn N, Cassano F, Berman E, et al (2023) Reflexion: Language agents with verbal reinforcement learning. In: Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:2303.11366

Tan JJY, Otto KN, Wood KL (2017) Relative impact of early versus late design decisions in systems development. Design Science 3:e12. https://doi.org/10.1017/dsj.2017.13

Tang J, Qin J, Thorat K, et al (2024) HiVeGen: Hierarchical LLM-based verilog generation for scalable chip design. URL https://arxiv.org/abs/2412.05393, arXiv preprint arXiv:2412.05393 [cs.AR]

Tang J, Qin J, Xu N, et al (2025) MAHL: Multi-agent LLM-guided hierarchical chiplet design with adaptive debugging. In: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), https://doi.org/10.1109/ICCAD66269.2025.11240973

Tian X, Du W, Yang S, et al (2026) From text to simulation: A multi-agent LLM workflow for automated chemical process design. In: Proceedings of the AAAI Conference

on Artificial Intelligence, https://doi.org/10.1609/AAAI.V40I35.40215

Tobin J, Fong R, Ray A, et al (2017) Domain randomization for transferring deep neural networks from simulation to the real world. URL https://arxiv.org/abs/1703.06907, arXiv preprint arXiv:1703.06907

Wang G, Xie Y, Jiang Y, et al (2023) Voyager: An open-ended embodied agent with large language models. In: Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:2305.16291

Wang H, Wang J, Zou B, et al (2026a) Coding agent is good as world simulator. URL https://arxiv.org/abs/2605.14398, arXiv preprint arXiv:2605.14398 [cs.AI]

Wang L, Ma C, Feng X, et al (2024a) A survey on large language model based autonomous agents. Front Comput Sci 18:186345. https://doi.org/10.1007/s11704-024-40231-1

Wang Q, Xie Y, Yang T, et al (2026b) Human-enhanced loop modeling (HELM): Agent-based finite element modeling of concrete bridge barriers. URL https://arxiv.org/abs/2606.12025, arXiv preprint arXiv:2606.12025 [cs.CE]

Wang S, Scells H, Zhuang S, et al (2024b) Zero-shot generative large language models for systematic review screening automation. URL https://arxiv.org/abs/2401.06320, arXiv preprint arXiv:2401.06320

Wang W, Ma X, He Y, et al (2026c) A lightweight multi-agent framework for automated concrete barrier design. URL https://arxiv.org/abs/2606.12040, arXiv preprint arXiv:2606.12040 [cs.SE]

Wang Y, Xu X, Li Y, et al (2026d) When search becomes memory: Turning robot design trials into transferable skills. URL https://arxiv.org/abs/2605.25832, arXiv preprint arXiv:2605.25832 [cs.RO]

Wang Z, Gao J, Fu W, et al (2025) AnalogSAGE: Self-evolving analog design multi-agents with stratified memory and grounded experience. URL https://arxiv.org/abs/2512.22435, arXiv preprint arXiv:2512.22435 [cs.AR]

Wang Z, Ma Y, Schmidt CR, et al (2026e) MDForge: Agentic molecular dynamics pipeline design under sparse simulator feedback. URL https://arxiv.org/abs/2606.12916, arXiv preprint arXiv:2606.12916 [q-bio.BM]

Wei J, Wang X, Schuurmans D, et al (2022) Chain-of-thought prompting elicits reasoning in large language models. In: Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:2201.11903

Xi Z, Chen W, Guo X, et al (2025) The rise and potential of large language model based agents: A survey. Science China Information Sciences 68(2). https://doi.org/

10.1007/s11432-024-4222-0, URL http://dx.doi.org/10.1007/s11432-024-4222-0

Xia Y, Mukherjee S, Xie Z, et al (2025) From selection to generation: A survey of LLM-based active learning. URL https://arxiv.org/abs/2502.11767, arXiv preprint arXiv:2502.11767

Yao S, Zhao J, Yu D, et al (2023) ReAct: Synergizing reasoning and acting in language models. In: International Conference on Learning Representations (ICLR), arXiv preprint arXiv:2210.03629

Yehudai A, Eden L, Li A, et al (2025) Survey on evaluation of LLM-based agents. URL https://arxiv.org/abs/2503.16416, arXiv preprint arXiv:2503.16416

Yu C, Deng C, Pinckney NR, et al (2026) Agentic hardware design as repository-level code evolution. URL https://arxiv.org/abs/2606.28279, arXiv preprint arXiv:2606.28279 [cs.AR]

Zhang C (2026) From reasoning to agentic: Credit assignment in reinforcement learning for large language models. URL https://arxiv.org/abs/2604.09459, arXiv preprint arXiv:2604.09459

Zhang H, Fan W, Gao X, et al (2026) PANDA: An LLM-enhanced performance-driven analog design framework bridging design intent and layout generation. URL https://arxiv.org/abs/2606.15052, arXiv preprint arXiv:2606.15052 [cs.AR]

Zhao W, Peña Queralta J, Westerlund T (2020) Sim-to-real transfer in deep reinforcement learning for robotics: A survey. URL https://arxiv.org/abs/2009.13303, arXiv preprint arXiv:2009.13303

Zhao Z, Xing Y, Xue T, et al (2026) AutoMS: Multi-agent evolutionary search for cross-physics inverse microstructure design. URL https://arxiv.org/abs/2603.27195, arXiv preprint arXiv:2603.27195 [cs.LG]

Zhou Z, Ye Y, Vadhavka SA, et al (2026) Learning cardiac electrophysiology digital twins through agentic discovery of hybrid structure. URL https://arxiv.org/abs/2606.18154, arXiv preprint arXiv:2606.18154 [eess.SP]

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- ESM2.csv
- ESM1.csv
- ESM3.zip