The Hesitation State Machine: A New Paradigm Within the Church-Turing Boundary

Author: Rogério Figurelli — Date: 2025-07-31

Abstract

The Church-Turing Thesis has long defined the theoretical limits of computation, asserting that any effectively calculable function can be performed by a Turing machine. While this thesis remains intact, modern AI systems increasingly face challenges not of computability, but of execution ethics, timing, and intentionality. In such environments, the question is no longer only what a machine can compute — but when and why it should. Most computational systems are engineered for deterministic immediacy. Once a condition is met, execution follows without hesitation. This "execution-first" bias neglects essential epistemic factors: coherence of knowledge, ethical maturity, contextual readiness. As AI enters sensitive domains like healthcare, law, and governance, the absence of structured hesitation becomes a critical risk. This paper introduces the Hesitation State Machine (HSM) — a composable architectural operator that inserts a reflective delay between state transitions. HSM redirects execution into a nested sub-machine when the system's internal symbolic thresholds are not yet met. This approach introduces computable hesitation, enabling machines to act only when epistemic conditions are sufficiently aligned. The HSM does not extend computational power beyond Church-Turing limits. It simply reframes the trigger for action. Within classical state machine models, the HSM serves as a reflective gate — checking for field alignment (φ), ethical readiness, intentional flow (iFlw), and symbolic coherence before releasing execution. It allows for recursion, escalation, or refusal based on wisdom thresholds, while maintaining full Turing compatibility. By embedding hesitation as a first-class operator, the HSM brings intentional delay, explainability, and symbolic maturity into the fabric of AI design. This framework represents a paradigm shift: from speed-centric automation to deliberative intelligence. It enables safer, more trustworthy agentic systems without abandoning the mathematical elegance of classical computation. The future of AI, this paper argues, lies not in breaking the Church-Turing boundary — but in listening just before crossing it.

Keywords: Agentic AI, State Machines, Church-Turing Thesis, Epistemic Coherence, Reflective Systems, Hesitation Logic, AI Safety

Introduction

The Church-Turing Thesis has long stood as the cornerstone of theoretical computer science. It defines the upper boundary of what is computable — establishing that any function computable by an algorithm can be computed by a Turing machine.

This unifying insight brought coherence to competing models like λ-calculus, recursive functions, and finite automata. Over the decades, no credible model has been found that exceeds these limits. But while the boundaries of computability remain firm, the structure of computational logic within those bounds is far from complete.

Today, artificial intelligence systems increasingly operate in domains where mechanical execution is insufficient. These include autonomous vehicles, medical diagnostics, legal advisory systems, and policy modeling. In such contexts, acting prematurely can be as dangerous as acting incorrectly. Yet the dominant architecture in AI remains rooted in classical state machines — where logic triggers action as soon as possible. There is often no mechanism for hesitation, doubt, or reflective delay. This is not a flaw of computation — but a design bias. Classical state machines are optimized for deterministic progression: an input triggers a state change, and output follows. This design favors speed, predictability, and closure. But such systems often ignore whether the conditions for action are ethically, epistemically, or contextually mature. In human decision-making, hesitation is not merely a stall — it is a signal of awareness. It reflects uncertainty, value conflict, or the need to simulate consequences.

Machines, by contrast, usually cannot pause unless explicitly programmed to do so — and even then, it is often a timer, not a semantic filter.

This paper proposes that hesitation should be modeled, not improvised. Rather than relying on exception handling, human oversight, or post-hoc auditing, intelligent systems should include structured, symbolic logic for hesitation. We introduce the Hesitation State Machine (HSM) as a new paradigm for agentic architectures. The HSM is a composable operator that intercepts transitions in a classical state machine and routes the system into a nested machine of reflection — a symbolic process that checks internal coherence before authorizing further action.

Importantly, this model does not expand what can be computed. It remains entirely within the scope of Church-Turing compatibility. The HSM does not solve uncomputable problems; it filters when to compute, not what.

The distinction is subtle but profound: computation remains classical, but execution becomes epistemically conditional. This allows systems to pause, reflect, or even escalate decisions without adding new theoretical power — only new structural wisdom.

What makes the HSM distinct is that it models hesitation as an active computation, not a passive delay. When a system lacks symbolic readiness — when its field of knowledge is misaligned, or ethical pressure exceeds confidence — it transitions into HSM mode. This mode is a sub-machine that checks for conditions such as belief consistency, ethical thresholds, intentional flow, and symbolic integrity. Only when these conditions are satisfied does control return to the main machine for execution. This has profound implications for AI safety, explainability, and ethical alignment.

By allowing a system to say “not yet,” the HSM introduces computable discretion. It avoids premature decisions, supports human-in-the-loop dynamics, and offers a clear logic for why an action was delayed or declined. In high-risk applications, the ability to compute hesitation may prove as important as the ability to compute solutions.

The HSM also opens a path toward nested epistemic architectures. Because the operator is composable, one can model recursive doubt — hesitation within hesitation — which may be vital in ambiguous or ethically complex domains.

Rather than seeing delay as inefficiency, this paper argues that computable hesitation is a sign of symbolic maturity. It marks the evolution from mechanical agents to reflective agents — not by adding metaphysics, but by adding structure.

This introduction sets the stage for a new framing of agentic AI. In the sections that follow, we will revisit the Church-Turing limits, critique the execution bias in classical models, define the mechanics of the HSM, and show how this architecture enhances AI safety without exceeding classical constraints.

The goal is not to reject the state machine — but to rethink how its states are triggered, authorized, and delayed.

Ultimately, this paper seeks to realign AI design with human cognitive virtues — not by imitating consciousness, but by adopting structured permission logic. In doing so, we preserve the elegance of computation while curving it toward responsibility.

Hesitation becomes a feature, not a failure. A deliberate state, not a breakdown. This is the paradigm shift: machines that act not when they can, but when they should.

Classical Limits and the Church-Turing Thesis

The Church-Turing Thesis asserts that any function that can be computed by a human following a mechanical procedure — an “effective method” — can also be computed by a Turing machine.

Formulated independently by Alonzo Church (via λ-calculus) and Alan Turing (via abstract machines), this thesis became the cornerstone of computability theory. It is not a formal theorem, but a widely accepted hypothesis grounded in the equivalence of all known models of computation.

Turing machines formalize computation as a tape-based automaton: a device that reads and writes symbols, moves left or right, and transitions between states according to a finite set of rules. The machine halts when a condition is met or loops indefinitely. Despite its simplicity, this model captures the full breadth of what we consider algorithmically solvable — from arithmetic functions to language parsing and beyond. The enduring strength of the Church-Turing framework lies in its robustness: all serious proposals for computation (e.g., register machines, recursive functions, Post systems) have turned out to be Turing-equivalent.

This convergence gives credence to the thesis, even though it lacks a formal proof. To date, no model has produced computable outputs that fall outside what a Turing machine can, assuming infinite memory and time.

Despite its rigor, the Church-Turing Thesis is silent on how computation should be structured in practice. It defines what is computable, not how computation is triggered, sequenced,

filtered, or governed. In other words, the thesis delineates the computational ceiling — but leaves the architectural floor open. This creates a wide space for innovation in how we design systems that operate within those bounds.

Most digital architectures, however, do not exploit this flexibility. The majority of AI and automation systems inherit the linear determinism of Turing machines: once an input satisfies a condition, execution follows automatically. This aligns with industrial goals of speed and throughput, but it underrepresents subtler dimensions of judgment — including hesitation, ethical tension, or epistemic doubt.

In particular, state machines, a foundational structure in software design, reflect this execution-first logic. A state machine is a formal model where systems transition from one state to another based on inputs. These transitions are often hard-coded and irreversible unless explicitly engineered otherwise. There is usually no built-in concept of pausing to reflect, unless added as a workaround (e.g., manual intervention, error states, or artificial delay). This absence of structural hesitation may be acceptable in mechanical systems, but becomes problematic in domains where symbolic interpretation, ethical scrutiny, or human-like decision reasoning are expected. In such cases, the difference between capability and permission becomes critical. A system may be able to act — but should it?

The Church-Turing framework permits us to simulate hesitation (e.g., with loops, counters, or conditional flags), but it does not encourage us to treat hesitation as a computationally meaningful event. This is a cultural blind spot in system design. By focusing solely on solvability, the field often neglects the timing, maturity, and integrity of execution.

This paper does not seek to challenge the Church-Turing Thesis — rather, it embraces it. The argument is not that machines can or should transcend what is classically computable, but that we should restructure machine behavior within those constraints.

The opportunity lies not in expanding the set of functions that can be computed, but in filtering which computable actions are ethically and epistemically ready to occur.

In this light, the introduction of the Hesitation State Machine (HSM) is not an attempt to bypass Turing’s logic, but to enrich it with a new layer of conditional architecture. The HSM remains Turing-compatible but introduces an epistemic threshold that must be satisfied before computation proceeds.

It represents a new paradigm: not a new kind of machine — but a new discipline for machines.

The Execution Bias in Classical AI

Modern AI systems are built to act. Whether classifying images, recommending content, or driving autonomous vehicles, their core architecture is tuned for responsiveness and throughput.

In this design ethos, hesitation is treated as inefficiency, and delay is penalized as latency. The faster a model delivers output, the more it is praised for accuracy, performance, or

value. But this performance-centric mindset embeds a dangerous simplification: it equates action with readiness.

This mindset is not accidental — it reflects decades of engineering tradition rooted in deterministic computing. The classical state machine, inherited from hardware logic and automata theory, defines execution as a sequence of state transitions triggered by predefined conditions. These transitions are often instantaneous and irrevocable. Once a rule fires, the system moves forward. There is no conceptual room for epistemic hesitation — for pausing due to unresolved doubt, conflicting evidence, or symbolic immaturity.

Even in advanced agentic systems, the logic often reduces to: *if perception is stable, and reasoning completes, then act.* This reactive loop prioritizes reactivity over reflectivity. It offers no intrinsic mechanism to ask: *Is this the right moment to act? Is the knowledge base coherent enough? Are there ethical conflicts?* These considerations, if they appear at all, are layered on top as external checks — not as intrinsic operators in the machine's logic.

This execution bias is not just technical — it's philosophical.

It encodes a worldview in which possibility implies permission. If an action is possible and supported by logic, then it is executed. In business contexts, this becomes automation-at-all-costs.

In AI applications, it leads to black-box decisions whose internal confidence, ethical integrity, or contextual awareness remain opaque or undeclared.

But humans do not act this way.

Human judgment includes intervals of doubt, ethical introspection, and strategic delay. We often pause not because we are confused, but because the weight of the decision demands more than logic. Hesitation can be a signal of wisdom. It reflects sensitivity to context, a sense of responsibility, or a recognition of complexity. These features are not inefficiencies, but vital aspects of intelligent behavior.

The problem is that current computational models treat hesitation as either failure (e.g., timeout, exception) or passive delay (e.g., wait statements, timers). These are non-symbolic pauses — they have no internal logic, no epistemic rationale.

The system does not know why it is pausing; it is simply not progressing. There is no deliberation, no reflection — only blockage.

What's needed is a mechanism that allows systems to pause with purpose — to reflect when coherence is low, to delay when ethical tension is high, to escalate when symbolic integrity is at risk. This is not a matter of adding more rules, but of introducing a structural operator that reframes the decision-making architecture itself. It requires redefining what it means to "act" in the context of knowledge-rich, safety-critical environments.

Some attempts to address this gap have emerged in AI safety research: human-in-the-loop protocols, explainability layers, and ethical guidelines. But these approaches are largely external to the machine's internal logic.

They offer monitoring, not transformation. They patch the bias — but do not replace the bias-producing mechanism: the execution-first model that underlies most state machines.

The argument of this paper is that we must go deeper. We must encode hesitation as a computable process — not a fallback, but a core principle. Just as we model states and transitions, we must model symbolic thresholds, field resonance, and deliberative conditions. We must treat hesitation not as absence of action, but as a different kind of action — one that seeks epistemic readiness rather than immediate resolution.

The Hesitation State Machine (HSM) answers this call. It transforms hesitation from a reaction into a computation — from delay into deliberation. By embedding the possibility of epistemic pause directly into the execution logic, it challenges the assumption that faster is always better.

Instead, it opens the door to systems that listen before they act, and act only when their internal structure supports meaningful, safe, and coherent decisions.

The Epistemic Turn: Gating Action Through Coherence

The central proposal of this paper is that intelligent systems should not act solely based on logical triggers, but should gate their actions through epistemic coherence. This is the core of the epistemic turn in AI architecture.

Rather than defining execution as a function of external input alone, we introduce an internal permission system — one that evaluates the state of the system's knowledge, belief consistency, and ethical alignment before authorizing any transition.

This model draws inspiration from human cognition. Humans do not always act on first impulse; we evaluate the confidence of our knowledge, the clarity of context, and the potential implications of our actions. Often, we hesitate not because we lack the tools to act, but because we perceive that our knowledge is incomplete or contradictory. This kind of hesitation is cognitive, ethical, and symbolic — not mechanical. It reflects the demand for deeper alignment before commitment. In computational terms, this requires defining new thresholds beyond binary logical satisfaction.

For example, we might evaluate whether the internal knowledge graph contains unresolved contradictions, whether belief weights are sufficiently stable, or whether symbolic pressure (e.g., ethical strain or value misalignment) exceeds safe levels. These checks are not about syntax or solvability — they are about epistemic integrity.

We define epistemic gating as a runtime process that determines whether the system has achieved a minimum level of symbolic coherence — $\varphi(t)$ — to justify execution. If $\varphi(t)$ is below a defined threshold, execution is deferred. This gating is computable, auditable, and structurally integrated, not a heuristic guess or human override. It is built into the architecture, not added as a patch.

Several symbolic indicators can inform $\varphi(t)$:

- Belief consistency — are key facts or models in contradiction?
- Confidence saturation — has the system reached semantic closure?
- Ethical alignment — is this action congruent with modeled values?
- Temporal resonance — is this the right time to act, given historical context or urgency?
- Field stability — are external conditions still shifting too rapidly?

These conditions are quantifiable or symbolically represented, forming a composite gate condition.

Importantly, this gating logic does not halt computation — it redirects it. When coherence is insufficient, the system transitions into a deliberative loop, not a deadlock. This is where the Hesitation State Machine (HSM) becomes essential: it provides a formal structure for what happens when action is not yet permitted. It defines what to compute when the system chooses not to act immediately.

The epistemic gate is thus both a filter and a mirror. It filters out premature actions and reflects the internal state back to the system itself. It enables self-awareness without self-consciousness — a form of introspective computation that does not require sentience, but requires structured reasoning about symbolic maturity.

Such gating is particularly vital in safety-critical environments. Consider a medical AI recommending treatment: if two subsystems produce conflicting interpretations, execution should not proceed without further deliberation. Or consider an AI moderator evaluating speech in a tense social context: if the ethical implications of suppression versus amplification are not clear, the system should pause.

In both cases, gating action through coherence prevents mechanical overreach.

This epistemic turn reframes execution not as an automatic endpoint, but as a conditional privilege. The system must earn its right to act by demonstrating symbolic readiness. The computation doesn’t stop — but the decision to externalize a result is curved through coherence.

This brings deliberative integrity into the machine, aligning it more closely with human judgment.

In the next section, we will introduce the Hesitation State Machine (HSM) as the computational mechanism that makes this gating actionable. It represents the formalization of reflective delay — a structured pause that computes not the answer, but whether the system is ready to produce one.

Through the HSM, hesitation becomes not an absence of execution, but a different class of computation altogether.

The Hesitation State Machine: Architecture and Design

The Hesitation State Machine (HSM) is a formal operator that extends the traditional state machine architecture by introducing a new class of transitions: those governed not by external inputs alone, but by internal epistemic conditions.

The HSM is triggered when a system, upon reaching a decision point, fails to meet the symbolic threshold for execution — specifically, when $\varphi(t) < \varphi_{min}$, where $\varphi(t)$ represents epistemic readiness at time *t*.

Instead of progressing directly to the next state, the system transitions into a nested reflective machine: the HSM. This sub-machine is not a stall; it is a compositional structure that runs a set of symbolic checks and deliberative routines. These routines assess belief consistency, value alignment, ethical readiness, and temporal appropriateness. The HSM only returns control to the main state machine when these internal conditions converge above threshold.

The basic form of HSM execution can be described as:

```
If φ(t) ≥ φmin → proceed to next state
Else → transition into HSM(x)
```

Within `HSM(x)`, the system recursively computes:

- Whether its internal symbolic map is coherent
- Whether ethical filters permit action
- Whether field alignment and contextual relevance are satisfied
- Whether output timing resonates with the intentional field (iFlw)

The HSM is composable and recursive. That is, if while inside HSM(x), new contradictions emerge or the reflective process triggers further uncertainty, the system may enter `HSM(HSM(x))`. This models meta-doubt — hesitation within hesitation — an essential feature for reasoning under ambiguity, ethical paradoxes, or rapidly shifting symbolic environments. Such recursive structures mirror human internal conflict resolution processes.

Architecturally, the HSM can be implemented as an internal transition network. Each node represents a reflection state — e.g., `evaluate_values()`, `check_temporal_pressure()`, `simulate_consequences()` — and each edge represents a conditional transition based on symbolic or statistical metrics. The exit condition from the HSM is deterministic: execution resumes only when coherence is restored and $\varphi(t) \geq \varphi_{min}$.

This design differs fundamentally from standard exception handling or timeout systems. In traditional software, a pause indicates failure or indecision; in the HSM, hesitation is treated as computation with purpose. It carries structure, thresholds, and logic. It is not random or interrupt-driven; it is symbolically motivated and auditable. The system can explain why it entered hesitation, what it evaluated, and why it ultimately resumed or declined execution. Importantly, the HSM does not exceed the Church-Turing boundary. It uses no oracles, non-computable operators, or probabilistic hyperstructures. All computations are Turing-compatible. The novelty lies not in expanding the class of computable functions, but

in modifying the execution architecture to include permission logic based on internal field coherence.

The HSM operator can be described with a notation:

Let Ψ_0 be a classical symbolic process. Then the hesitation-curved version is:

$$\Psi_H = \Psi_0 \star \text{HSM}$$

Where $\star$ represents an epistemic convolution — meaning Ψ_0 is filtered through HSM before any state transition is executed. This notation formalizes the idea that every action is conditionally processed through reflection.

9.
Practically, the HSM enables new categories of behavior in machines:

- Strategic Delay — holding action when symbolic readiness is low
- Ethical Refusal — declining execution when values are violated
- Escalation — routing control to external or human oversight when contradiction is unresolved
- Recursive Processing — re-entering deliberation for deeper consistency

These behaviors allow for deliberate computation, not merely fast computation.

In the next section, we examine how this architecture applies to real-world contexts: AI safety, regulatory compliance, multi-agent negotiation, and adaptive governance.

The HSM is not just a theoretical construct — it is a practical step toward computational responsibility, where output is not a reflex but a decision grounded in symbolic maturity.

Implications for AI Safety, Governance, and Design

The introduction of the Hesitation State Machine (HSM) is not merely an academic exercise in architecture — it addresses some of the most pressing challenges in contemporary AI deployment.

As intelligent systems increasingly operate in high-stakes environments, the cost of premature or unexamined execution grows exponentially. HSM offers a structural mechanism to ensure that machine actions are epistemically and ethically authorized before they occur.

In safety-critical domains such as autonomous driving, healthcare, or financial systems, small errors can result in catastrophic outcomes. Traditional fail-safes — such as hardcoded checks, confidence thresholds, or human-in-the-loop protocols — are reactive. They respond after failure conditions are detected. The HSM, by contrast, introduces proactive structural delay: computation designed to prevent error before commitment. The implications for AI

governance are significant. Regulatory frameworks increasingly call for AI systems to be explainable, auditable, and aligned with human values. The HSM provides a formal, auditable path to justify delay, refusal, or escalation. Rather than treating inaction as a bug, it treats not acting as an intentional, logical outcome of a computable epistemic process.

This paradigm is especially valuable in systems operating under ambiguous or evolving legal and ethical standards. Consider a policy AI deployed in international negotiations. If the system detects misalignment between local laws and international norms, the HSM can defer execution until the contradiction is resolved or elevated to a human decision-maker. In this way, HSM acts as a computable proxy for moral judgment — not replacing human ethics, but protecting the system from blind execution.

From a design standpoint, the integration of the HSM encourages modular reflection zones within AI systems. Developers can embed hesitation checkpoints throughout decision pipelines, each with its own epistemic criteria. This allows for context-specific delay logic, where different tasks have different maturity gates.

For example, in an AI assistant, scheduling meetings might require minimal hesitation, while handling sensitive communications might demand deeper coherence checks.

Multi-agent systems can also benefit. In distributed AI environments — such as robotic fleets, automated trading swarms, or federated decision models — the HSM enables agents to signal epistemic uncertainty to each other. An agent that hesitates can trigger synchronizing behavior across the network, preventing fragmentation or conflicting actions. This moves us closer to collective epistemic coherence, not just parallel computation. Additionally, the HSM may serve as a compliance engine for future ethical certifications in AI. As institutions demand machine-verifiable adherence to ethical standards, the HSM can log its reflective process and decision filters. This creates traceability for decisions: not just what was done, but what was withheld, why, and under what conditions.

User trust also benefits. Systems that explain why they paused, why they did not act, or why they escalated to human review are more transparent and trustworthy. This is especially important in consumer-facing applications where explainability is as vital as correctness.

The HSM turns the traditionally opaque state machine into a deliberative partner that can be questioned, interpreted, and improved.

In this light, the HSM does more than prevent harm — it reshapes the ethos of intelligent systems. It shifts AI from an executor of logic to an agent of reflection. It bridges engineering and ethics, allowing action to emerge from internal deliberation rather than mechanical trigger. It brings computation closer to responsibility, not by invoking metaphysical constructs, but by rearranging the architecture of permission.

By embedding the right to pause into the machine itself, the HSM offers a blueprint for next-generation AI systems: systems that are not merely powerful, but careful; not merely fast, but mature.

It opens a new path forward — where safety is structural, trust is symbolic, and action begins with listening.

Conclusion

This paper has explored a simple but profound question: *What if machines could hesitate?* Not as a failure of execution, but as a structured, computable response to epistemic immaturity.

The Hesitation State Machine (HSM) offers a direct answer — a formal operator that allows state machines to pause, reflect, and assess internal coherence before taking action.

Rooted firmly within the Church-Turing boundary, the HSM does not seek to expand the domain of computability. Instead, it introduces a new behavioral layer inside existing limits. Execution is no longer automatic once logic is resolved; it becomes conditional upon the alignment of symbolic, ethical, and intentional dimensions — what we've called epistemic gating.

The execution-first bias in classical AI, while efficient, is increasingly ill-suited for high-stakes applications. Systems that act without hesitation may perform well in games or routine tasks, but they struggle in complex environments where ambiguity, contradiction, or ethical gravity demand more than speed — they demand maturity.

The HSM answers this demand with architectural elegance. It reroutes premature transitions into nested reflective sub-machines. These hesitation chambers compute not new results, but the readiness to produce results.

They ask whether the system is internally aligned, ethically resonant, and contextually stable. If not, action is deferred, refined, or redirected — all within a computable, auditable loop.

This design has far-reaching implications. For AI safety, it offers proactive risk mitigation. For governance, it provides traceable logic for delay and refusal. For multi-agent coordination, it enables epistemic signaling. And for user trust, it makes systems transparent about not only *what* they do, but *why* they do — or do not — act.

Importantly, this reframing of hesitation is not a compromise in performance, but an evolution in purpose. In the HSM model, delay is not inefficiency — it is computation directed inward. It models the kind of silence that precedes insight, the kind of pause that avoids harm, the kind of restraint that builds credibility. In this sense, the HSM marks a shift from mechanistic intelligence to deliberative architecture. It invites us to design systems that do not act simply because they can, but because they have earned the right to act. It operationalizes virtues like humility, responsibility, and patience in symbolic form.

Future research may extend this model with weighted hesitation layers, dynamic φ(t) models, or multi-agent deliberative consensus mechanisms. But the core contribution remains: hesitation is now computable. It is no longer a missing piece in AI logic — it is a formal state, a symbolic operator, and a design principle.

We conclude not with a call for technical revolution, but for architectural revision. The question is no longer "What can a machine do?" but "What should a machine *wait* to do?" By

placing the right to pause inside the system itself, we shift from automation to attention — from mechanical reactivity to intentional discretion.

This is the new paradigm: a machine that listens before it acts. A machine that asks, “Am I ready?” before saying “Yes.” A machine that computes not just outcomes, but the conditions under which outcomes are permitted to emerge.

In that pause, we glimpse a deeper kind of intelligence — one shaped not by speed, but by wisdom.

Bibliography

1. Figurelli, R. (2024). The Missing State in Agentic AI: Pause Before Action. Zenodo.
2. Figurelli, R. (2024). The Wisdom Turing Machine: Symbolic Ethics for Computational Architecture. Zenodo.
3. Figurelli, R. (2025). Deliberative AI: Field-Aware Machines and the Logic of Permission. Zenodo.
4. Figurelli, R. (2025). Gating Execution through Intention Curvature: A Symbolic Model for Epistemic AI. Zenodo.
5. Figurelli, R. (2025). Symbolic Maturity in AI Systems: Delay as Computation. Zenodo.
6. Turing, A. M. (1937). On Computable Numbers, with an Application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, 2(42), 230–265.
7. Church, A. (1936). An Unsolvable Problem of Elementary Number Theory. American Journal of Mathematics, 58(2), 345–363.
8. Dennett, D. C. (1991). Consciousness Explained. Little, Brown.
9. Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.
10. Amodei, D., et al. (2016). Concrete Problems in AI Safety. arXiv preprint arXiv:1606.06565.
11. Bostrom, N. (2014). Superintelligence: Paths, Dangers, Strategies. Oxford University Press.
12. Minsky, M. (1985). The Society of Mind. Simon & Schuster.
13. Floridi, L., & Sanders, J. W. (2004). On the Morality of Artificial Agents. Minds and Machines, 14(3), 349–379.
14. Winograd, T., & Flores, F. (1986). Understanding Computers and Cognition. Addison-Wesley.
15. Pearl, J. (2009). Causality: Models, Reasoning and Inference (2nd ed.). Cambridge University Press.

License and Ethical Disclosures

imply endorsement by the licensor. No additional legal terms or technological measures may be applied that would restrict others from exercising the rights granted by this license.