Fluency Is the Feeling of a Claim

How emotions govern the reasoning loop — in people, and in the AI systems we now program with prompts

---

Abstract

We feel things while we think: the warm hum of *this is clever*, the cold prickle of *I'm not sure*. This article argues that such feelings are best understood not as verdicts about truth but as **instrumentation** — and, more strongly, as a **control layer** that governs how much we verify before we act. Drawing on the cognitive science of processing fluency and the Feeling of Rightness, it shows why the pleasure of cleverness is a signal usually worth *inverting* and the discomfort of doubt a signal usually worth *obeying*, then locates every confidence-versus-grounding state on a single map. It then turns to AI-assisted software work, where the stakes sharpen: a language model is a fluency generator decoupled from verification, and the move from one-shot prompting to looped, agentic interaction is — provably — the move from a finite automaton to a Turing-complete computer. That framing yields the central claim: the documented failure modes of autonomous AI agents are the negative image of the human emotional system, because both are feedback loops whose correctness depends on contact with reality — an open loop has no natural *entropy sink*, no way to shed accumulated error except by touching the world — and emotions are the **governors** that force that contact. The practical upshot is that engineering a reliable prompt loop is the work of hand-installing, in software, the governors that doubt and its kin supply in a person for free.

How to read this

The argument builds in three movements — the cognitive science, the computational reframe, and the synthesis — and closes with practice. Read straight through, or jump to what you need:

1. **Feelings are information** — why metacognitive ease gets read as truth (the fluency–truth effect).
2. **The Feeling of Rightness** — the mechanism: fluency sets the brake on how much you verify.
3. **Why cleverness inverts** — Kernighan's law as a corollary; clever-intricate versus elegant-simple.
4. **Why doubt is obeyed** — doubt as an honest low-evidence reading; the illusion of explanatory depth.
5. **The four epistemic positions** — a confidence × grounding map, and the one genuinely dangerous quadrant.
6. **The LLM amplifier** — why an AI assistant industrializes these failure modes rather than inventing new ones.
7. **Prompting is programming** — the one-shot prompt as a finite automaton; the loop as a Turing-complete computer.
8. **The behavioral isomorphism** — emotions as governors, and the agent-failure bestiary as their negative image *(the core table lives here)*.
9. **Turning feelings into practice** — what to actually do at the keyboard.

A closing **Coda** returns to the two feelings the article opens on.

The argument in one line

Those feelings are not verdicts about whether you are right. They are **instrumentation** — gauges that report on the state of your reasoning, not on the state of the world — and the discipline of working well, especially in software and especially alongside AI assistants, is learning to read the gauges rather than believe them.

This matters because the two most diagnostically useful feelings point in *opposite* directions. The pleasure of cleverness is a signal you should usually **invert**. The discomfort of doubt is a signal you should usually **obey**. They feel like the same kind of thing — internal states with a confident emotional charge — but one is typically a false positive and the other a true negative. Naming that asymmetry is where the argument starts; by the end, those two feelings turn out to be two members of a larger control system that any reliable reasoning loop, human or machine, must implement somehow.

---

1. Feelings are information, and that is exactly the problem

There is a well-developed line of research in cognitive psychology on *metacognitive experiences* — the sensations of ease or difficulty that ride along with mental work. The central finding, associated with Norbert Schwarz's "feelings-as-information" account, is that people treat these sensations as *data* about the content of their thoughts, often without realizing they are doing so. Ease feels like truth. Difficulty feels like falsity or risk.

The cleanest demonstration is the **fluency–truth effect**. When a statement is easier to process, people are more likely to judge it true — and the manipulations that produce this are almost insultingly superficial. Printing a claim in higher color contrast, setting it in a cleaner font, making it rhyme, or simply repeating it all increase the rate at which people endorse it as true, independent of whether it is. In one widely cited paradigm, statements shown in high contrast were judged true more often than the identical statements shown in low contrast (Reber & Schwarz, 1999; Unkelbach, 2007).

The unsettling part is that fluency is not a *stupid* cue. In ordinary environments, true things genuinely are encountered more often, explained more coherently, and recalled more easily, so ease has real **ecological validity** as a predictor of truth (Unkelbach, 2007). The heuristic earns its keep most of the time. That is precisely why it is dangerous: it is a generally-reliable signal that becomes catastrophically wrong in exactly the situations engineered to be fluent without being correct — polished prose, confident delivery, well-formatted code. The signal cannot tell the difference between *true* and *merely easy*.

---

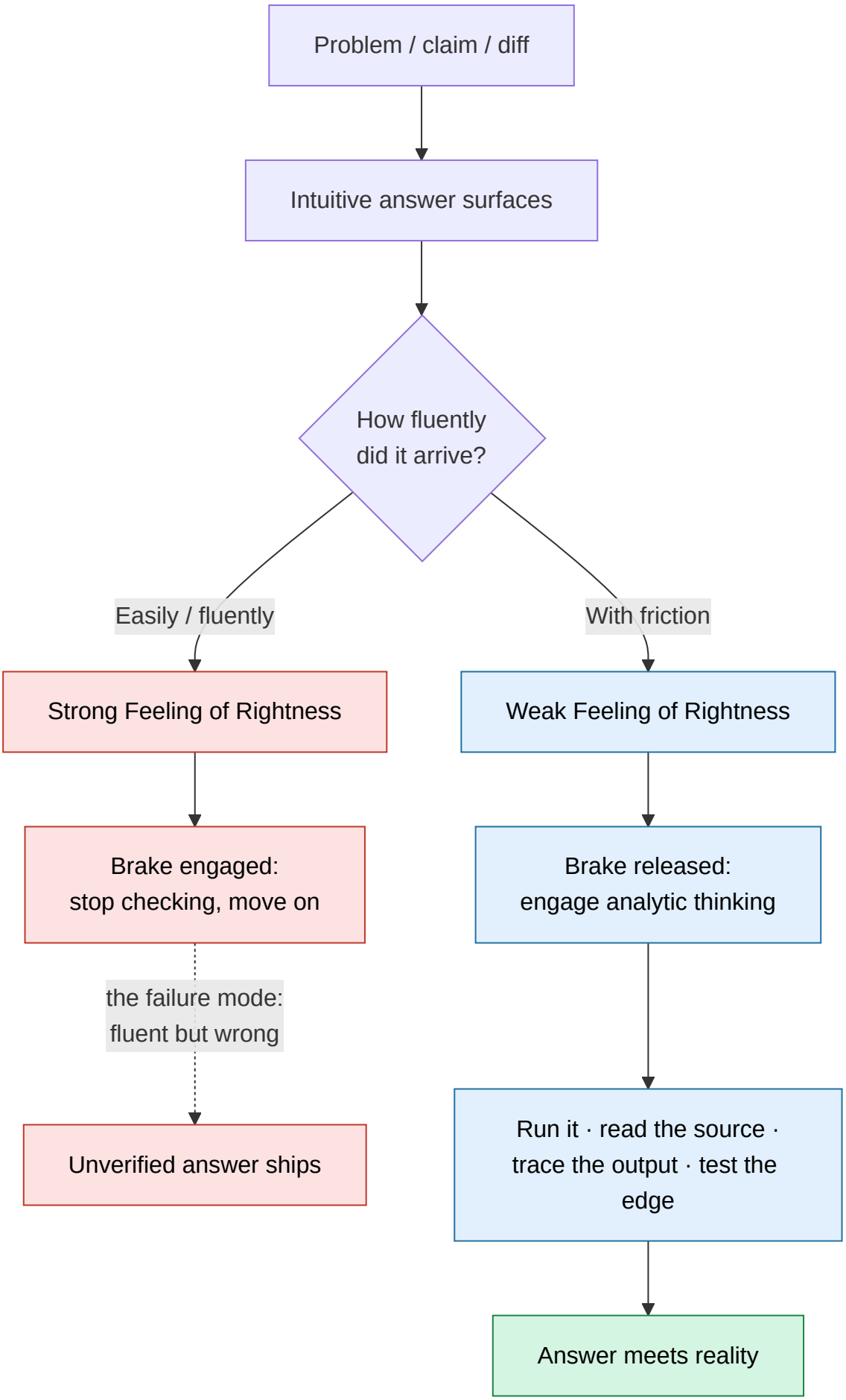
2. The mechanism: the Feeling of Rightness governs whether you keep thinking

The fluency effect would be a curiosity if it stopped at truth ratings. It does not. It controls **how much thinking you do next**.

Valerie Thompson's *Metacognitive Reasoning Theory* gives the mechanism a name: the **Feeling of Rightness (FOR)**. When an intuitive answer surfaces, it arrives bundled with a metacognitive sense of

how right it is. That feeling is not generated by checking the answer — it is generated largely by the *fluency* with which the answer came to mind. And the FOR then acts as a control signal: a strong FOR tells you no further analysis is needed and you move on; a weak FOR flags that something deserves a second look (Thompson, Prowse Turner, & Pennycook, 2011).

Empirically, this is robust. Answers that come fluently produce higher FOR; higher FOR predicts *less* rethinking time and a *lower* probability of changing the answer on reflection. The feeling of rightness is, functionally, the brake pedal on deliberate reasoning — and fluency is pressing it.


Problem / claim / diff
Intuitive answer surfaces
How fluently did it arrive?
Easily / fluently
With friction
Strong Feeling of Rightness
Weak Feeling of Rightness
Brake engaged: stop checking, move on
Brake released: engage analytic thinking
the failure mode: fluent but wrong
Unverified answer ships
Run it · read the source · trace the output · test the edge
Answer meets reality

The diagram makes the lever visible. Nothing in the loop measures whether the answer is *correct*. The thing that decides whether you verify is a feeling about how *smoothly* the answer arrived. That is the entire vulnerability, and both of the emotions we care about are interventions on this loop.

---

3. Why "feeling clever" should usually be inverted

Software engineering encoded this folklore decades before the psychology had a vocabulary for it. The canonical statement is Brian Kernighan's: “debugging is twice as hard as writing the code” in the first place — so if you write it as cleverly as you possibly can, you have, by construction, written something you are not sharp enough to debug (Kernighan & Plauger, *The Elements of Programming Style*, 1978).

Read through the FOR loop, the warning becomes mechanical rather than moral. The feeling of cleverness is a **reward response**, and it is calibrated to the wrong target. It fires on *intricacy* — novelty, density, a trick only you can currently see — because those properties are what make the solution feel impressive. But intricacy is precisely what *lowers* future fluency: for the next reader, for the debugger at 2 a.m., for you in six months, the clever solution is the disfluent one. So the cleverness reward is, structurally, a signal that you have optimized for a sensation that is anti-correlated with the thing you actually want.

A crucial distinction keeps this from becoming anti-intellectualism. There are two feelings that both register as pleasant and are easily confused:

	Clever-intricate	**Elegant-simple**
Source of the pleasure	"Only I can see how this works"	"Now anyone can see how this works"
Effect on future fluency	Lowers it (for everyone, including future-you)	Raises it
What it optimizes	The writer's ego in the moment	The reader's comprehension over time
Diagnostic question	*Why does this feel impressive?*	*Why does this feel inevitable?*
Read the signal as	A yellow flag — go simplify	Earned — but still verify behavior

The tell is the question *why does this feel good?* If the honest answer is "because it's subtle," that is the invert-me signal. If the answer is "because it made a hard thing obvious," that is a different and trustworthy pleasure — though even elegance is a feeling about your model, not a proof that the model is right.

---

4. Why doubt should usually be obeyed

Doubt is the mirror image, and it is the more honest of the two feelings. Where cleverness is a *false positive* — a strong FOR you have not earned — doubt is a *true negative*: a low FOR correctly reporting that you are running on claims you cannot presently vouch for. What the docs say. What you assume the function does. What you remember being true last release.

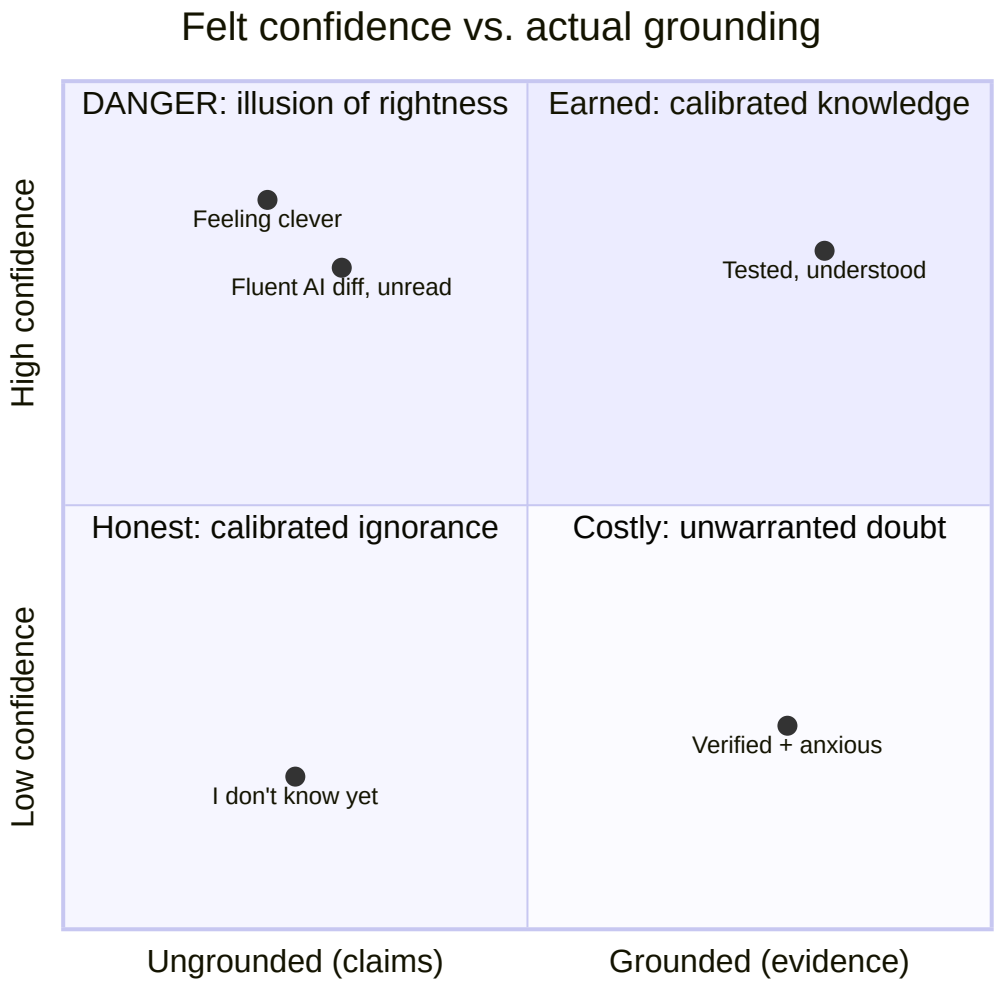
The feeling is accurate. The mistake people make is in how they *discharge* it. Doubt resolved by reassuring yourself — re-reading your own code until it looks fine, telling yourself it's probably correct — is doubt ignored with extra steps. Doubt resolved by *checking* — running it, printing the value, opening the actual source, writing the failing test — is doubt working as designed. The feeling is asking you to convert a **claim** into **evidence**, and only one of those two responses does that.

This is also where one of the most relevant biases lives: the **illusion of explanatory depth**. Rozenblit and Keil (2002) showed that people believe they understand how things work — zippers, bicycles, refrigerators — in far more detail than they actually do, and the illusion collapses the moment they are asked to produce a step-by-step explanation. Their *rated* understanding drops after they try and fail to explain. The felt sense of understanding runs ahead of the usable kind. Doubt, in this light, is the early-warning system for the gap between *I feel like I understand this code* and *I could actually explain what this line does and predict what it returns*. Suppressing the doubt keeps the illusion intact. Acting on it — trying to explain, and discovering you can't — is what punctures it.

---

5. A map of the four epistemic positions

If we treat the two underlying variables as independent — **how confident you feel** and **how grounded that confidence actually is** — we get a 2×2 that locates every state you can be in, and shows where each feeling is trying to move you.



The same four cells, with what produces each and what to do:

Quadrant	Feels like	Actually is	Produced by	Correct move
Earned (high confidence, grounded)	Calm certainty	Calibrated knowledge	Confidence that *followed* verification	Proceed; this is the goal state
DANGER (high confidence, ungrounded)	Cleverness, "obviously right," fluent ease	**The illusion of rightness**	Fluency, repetition, polish, automation bias	**Invert the signal.** Treat smoothness as a reason to check
Honest (low confidence, ungrounded)	Doubt, "I don't know yet"	Calibrated ignorance	Accurate read of a thin evidence base	**Obey the signal.** Go convert a claim into evidence
Costly (low confidence, grounded)	Anxiety, impostor unease	Unwarranted doubt	Habit, perfectionism, low self-trust	Notice you *did* the work; let the evidence settle it

The top-left **Danger** cell is the whole reason to write any of this down. It is where *feeling clever* lives, where a fluent unread diff lives, and where the illusion of explanatory depth lives. It is the only quadrant where your feeling and your reality actively disagree *in the dangerous direction* — and it is the one fluency reliably pushes you toward. Doubt's entire job is to move you leftward and downward out of false confidence, toward the honest cell, from which checking moves you rightward into earned knowledge.

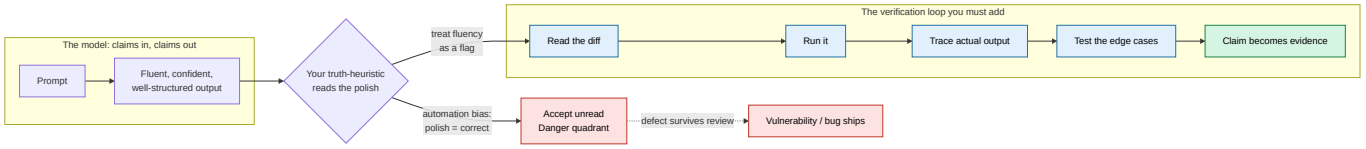
---

6. The LLM amplifier: a fluency generator decoupled from verification

Everything above is ordinary human cognition. Working with an AI coding assistant — Claude Code, Copilot, and their kin — does not introduce a new failure mode so much as it **industrializes the existing ones**, because a large language model is, almost by construction, a machine that produces the *feeling* of correctness without the *contact with reality* that would normally earn it.

Consider what the model optimizes for: fluent, confident, well-structured output. Those are exactly the surface properties that, per the fluency–truth effect, manufacture a high Feeling of Rightness — not in the model, but in *you*, the reader. A clean, plausible, well-commented diff trips your *this looks right* response by proxy. The polish is real; the correctness is a separate question the polish cannot answer. The model traffics in plausible **claims**, and its fluency borrows your own truth-heuristic against you.

The empirical picture is consistent with this. In a controlled study, developers with access to an AI assistant wrote **less secure** code than those without — yet were **more likely to believe** their code was secure (Perry, Srivastava, Kumar, & Boneh, 2022). The assistance raised confidence while lowering quality: a textbook excursion into the Danger quadrant. This sits inside the older and broader literature on **automation bias** — the tendency to over-trust automated output and discount contradictory evidence — where a systematic review found erroneous automated advice was followed at substantially higher rates when an automated recommender was present (Goddard, Roudsari, & Wyatt, 2012), an effect that the 2026 *International AI Safety Report* (§2.3.2) notes persists with current AI tools. Notably, some recent accounts find the effect can be *stronger* in experienced developers, as early correct suggestions build a trust that suppresses scrutiny of later ones.



The point of the diagram is the asymmetry between the two paths. The model's half of the system produces *claims* — fluent, plausible, structurally sound claims. It cannot produce *evidence about your system*, because it never ran against your system. The evidence loop is the part only you can close, and the feeling that tempts you to skip it — *this looks right, I can just accept it* — is the exact false-positive FOR that the model is unusually good at inducing. There is even a documented cousin of this in the explanation literature: readable AI explanations make users *feel* they understand a model's behavior while leaving them unable to predict it in new cases (Chromik et al., 2021) — the illusion of explanatory depth, delivered on demand.

---

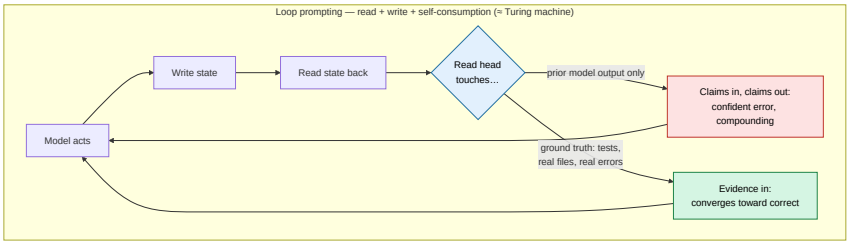
7. Prompting is programming; the loop is the computer

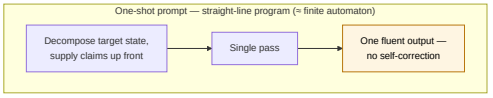
There is a reframe lurking under Section 6 that is worth making explicit, because it changes the verification loop from a piece of advice into a statement about *architecture*. What we have been calling "working with an AI assistant" is, increasingly, **programming** — just at a higher level of abstraction. And the distinction between a single prompt and a looped, agentic interaction turns out to be the exact distinction between two well-defined classes of computation.

Consider the two modes on their own terms.

A **one-shot prompt** is a straight-line program. You decompose a target state — the change you want made to a file, a dataset, a document, a reader's understanding — into a specification, supply the relevant claims and context up front, and receive a single output. There is no iteration and no feedback; what you get is a one-pass transformation of input into output, errors included, with no opportunity for the system to consume and correct its own work. This is not a loose analogy. A deterministic language model that conditions on a bounded-length string is formally equivalent to a **finite automaton** — expressive, but fundamentally *bounded* in the computations it can express (Schuurmans, 2023). The straight-line program is powerful and limited in exactly the way a circuit is.

Loop prompting — sometimes now called *prompt loop engineering* — is a different animal, and the difference is categorical rather than incremental. The moment the system can **read state, write state, and consume its own changes** on the next iteration, you have supplied the three ingredients of universal computation: read/write memory, iteration, and conditional control flow. The result is no longer a finite automaton. Augmented with an external read-write memory and an interaction loop, a large language model becomes **computationally universal** — able, in principle, to simulate any algorithm, i.e. Turing complete (Schuurmans, 2023; Schuurmans, Dai, & Zanini, 2024). The 2024 result is worth stating precisely, because it makes "prompting is programming" considerably less metaphorical: a universal Turing machine can be simulated by a Lag system of 2,027 production rules, and a single system prompt drives a specific production model, under greedy decoding, to apply all of them. Chain-of-thought sits on the rung between: by writing intermediate tokens it can read back, the model is already using a scratchpad as a primitive tape, which provably extends what it can compute (Feng et al., 2023; Merrill & Sabharwal, 2024). A loop that reads and writes and feeds on itself is, in the precise sense, a true computer.





Here is where the whole article's argument acquires teeth.

> **A Turing machine's correctness depends entirely on what its read head touches.**

The loop does not escape the fluency trap of the earlier sections — it **automates** it, in whichever direction you wire it. Point the loop's read operation at the model's own prior output, so that each turn ingests the last turn's fluent assertions as if they were findings, and you have built a computationally universal machine for generating *self-consistent, confident error* — drift and compounding mistakes wearing the polish that trips your truth-heuristic every iteration. Point the same read operation at **ground truth** — run the test and read the actual failure, open the real file, hit the live API, observe the real exception — and the identical architecture becomes a verification engine that converges toward correctness.

One caveat keeps this honest, and it matters because the surrounding formalism is itself fluent enough to trip the very heuristic this article warns about. Computational universality is an *expressivity* result: it says the loop can, in principle, express any computation — not that its output is correct. "Correctness depends on what you read" is a property of *feedback*, and it holds of a thermostat as much as of a Turing machine; universality is not what creates the dependency. What universality changes is the *ceiling*. A bounded automaton wired to its own output can only drift so far; a universal loop can amplify either fluent fiction or grounded truth without bound. So Turing-completeness does not manufacture the read-head problem — it removes the limit on how bad, or how good, it can get.

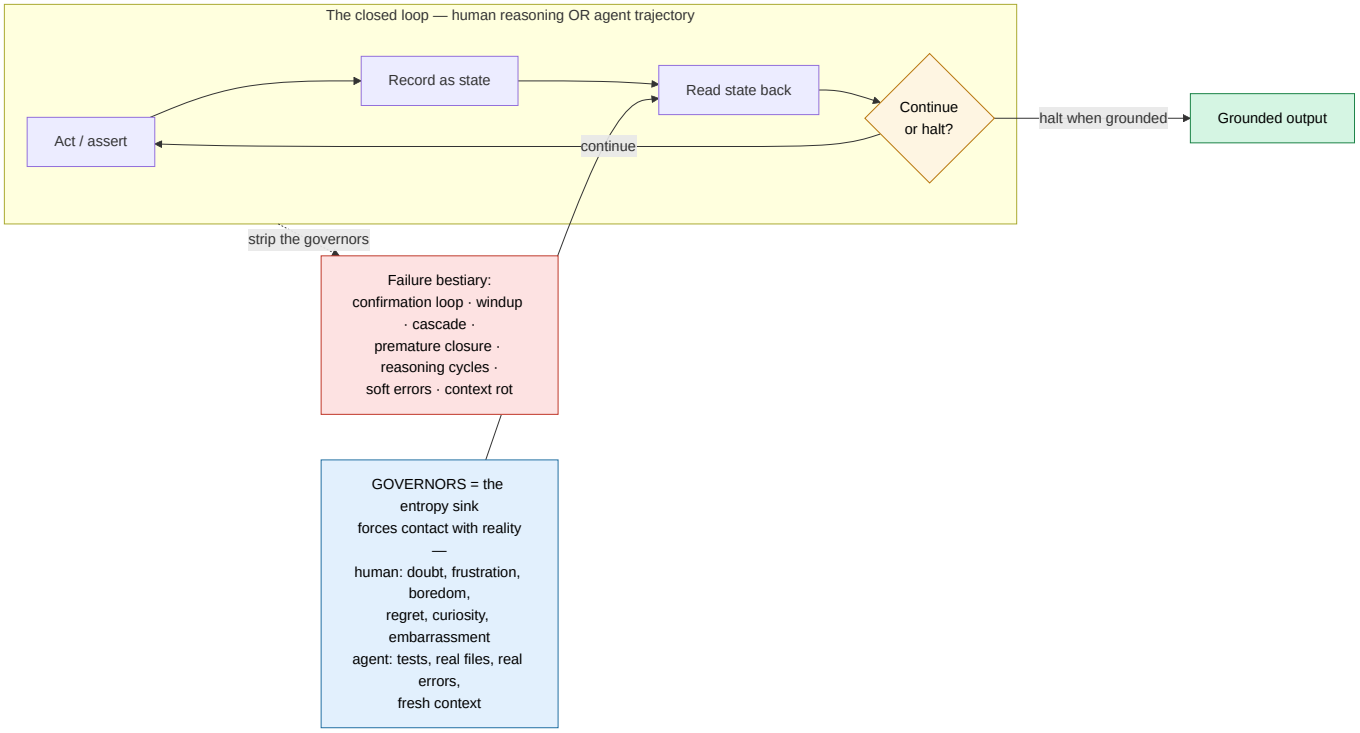
This is the Feeling-of-Rightness control loop from Section 2, lifted out of your head and rebuilt in software. The loop's stopping condition *is* the brake on analytic engagement; the question "what does the read head touch?" *is* the question of whether that brake is wired to fluency or to evidence. So the epistemic discipline that earlier sections framed as a personal habit becomes, in loop prompting, a literal **design parameter of the program you are writing**: where, in each iteration, does the loop make contact with reality? "Claims in, claims out" versus "evidence in" is no longer a warning about a single diff — it is the choice that determines whether your Turing-complete prompt loop computes the truth or computes an ever-more-confident fiction.

---

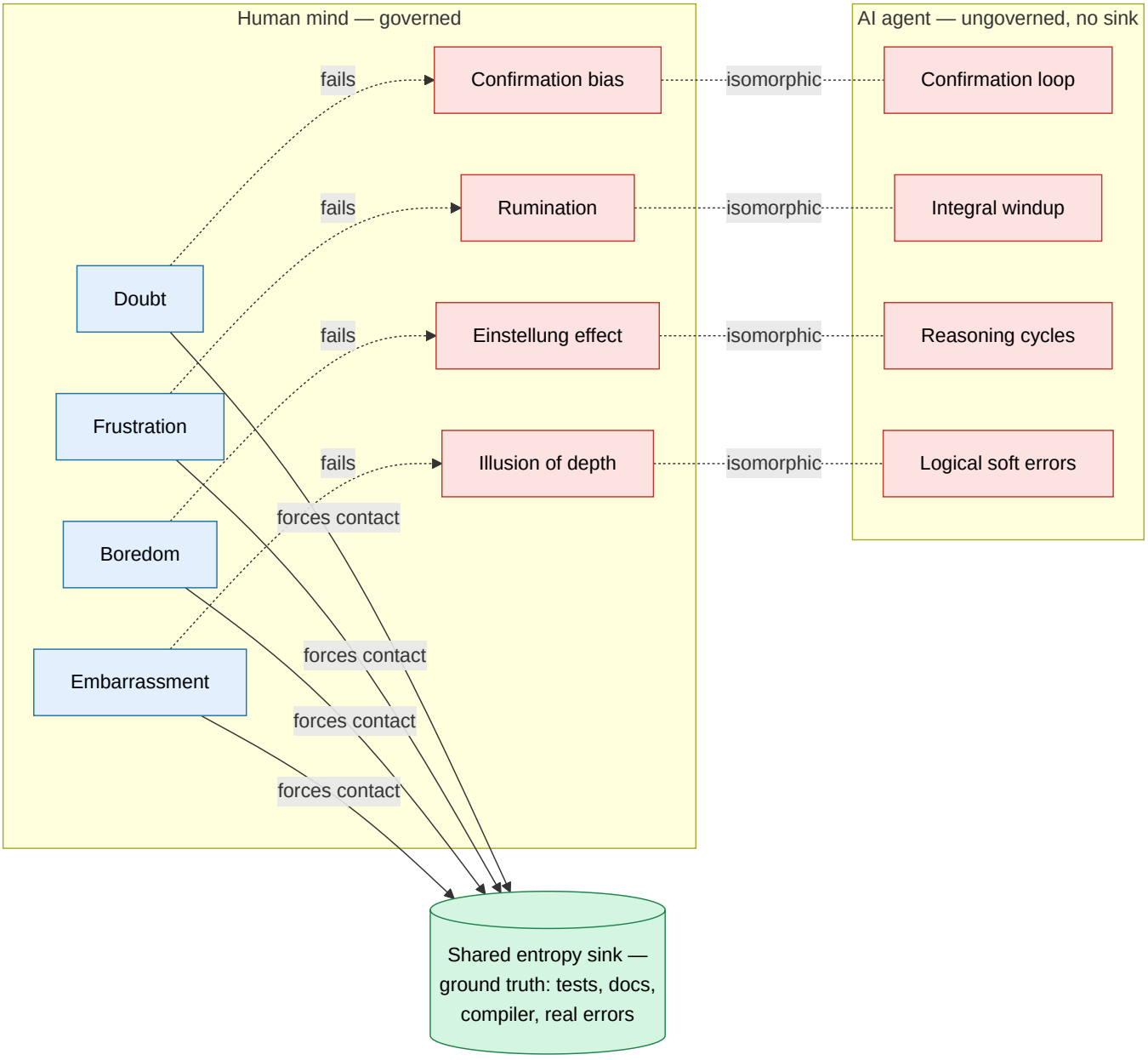
8. The behavioral isomorphism: a loop has the failure modes of a mind without governors

The previous section established that a closed loop is only as good as what its read operation touches. This section pushes on the consequence, because it explains something otherwise strange: the documented failure catalog of autonomous AI agents reads, almost line for line, like a catalog of human cognitive biases. That is not a coincidence or a cute analogy. Both are **feedback systems whose correctness depends on contact with reality**, and a feedback system with no external correction drifts in characteristic, nameable ways.

The organizing concept comes, fittingly, from the agent-systems literature itself: there is **no natural entropy sink** in an open loop (entropy-principle analysis of open agent systems, 2026). A self-referential process accumulates error with nowhere to export it unless something outside the loop forces a correction. In a healthy human reasoner, that "something outside the loop" is largely **affective**. The feelings this article has been treating as instrumentation — doubt and the cleverness-reward chief among them — are better understood as a *control layer*: evolved governors that periodically yank the cognitive loop back into contact with the world. The recognizable human biases are what reasoning looks like when a governor is weak, miscalibrated, or overridden. The agentic loop, by default, ships with **no governors at all** — which is exactly why its failure modes are the negative image of the human affective system.



The mapping below is the heart of the argument. Each row is a single control function viewed three ways: the **affective governor** that performs it in a person, the **named bias** that results in a person when that governor fails, and the **isomorphic agent-loop failure** that results in a machine that never had the governor to begin with. The first two rows are the pair this article opened with; the rest are the family they belong to. The diagram previews four of the mappings; the full table follows.



Each governor does double duty: working, it routes the loop to the entropy sink (green); failing, it leaves the human bias (red). The agent has the matching red failure with no governor and no sink to reach — which is why the two red columns are *isomorphic*.

Affective governor (what it monitors)	**Human failure when it's absent — *psychological term***	**Isomorphic agent-loop failure — *machine-intelligence term***
Doubt — *is this a claim or evidence?*	Confirmation bias; **illusory-truth effect** via self-repetition (Hasher, Goldstein & Toppino, 1977)	**Self-reinforcing confirmation loop** — the agent treats its own prior output or memory as ground truth when it is wrong (agent-memory write-manage-read analyses, 2025–26)
Cleverness, read as a warning — *am I optimizing a proxy for the goal?*	**Goodhart's law** — a measure that becomes a target ceases to be a good measure (Strathern, 1997)	**Reward hacking / specification gaming** — optimizing the proxy objective while diverging from intent
Frustration — *is this strategy converging?*	**Rumination** — recirculating without resolution (Nolen-Hoeksema, 1991)	**Integral windup** — non-converging retry loops, the classical control-theory signature of an unbounded accumulating corrector (control-theoretic framings of agentic systems, 2026)
Regret / the sunk-cost wince — *should I abandon this path?*	**Escalation of commitment** — throwing good resources after a failing course (Staw, 1976)	**Error propagation / cascading failure** — an undetected early mistake compounds through the trajectory; agents are observed persisting with a failed action rather than revising the state that produced it, until the task collapses (AgentErrorTaxonomy, 2025; OSWorld via SHIELDA §3.1.29, 2025)
Lingering doubt / curiosity — *have I checked enough to stop?*	**Premature closure**; high **need for cognitive closure** (Webster & Kruglanski, 1994)	**Stopping too early** — the loop terminates on partial success signals and shallow outcome checks, with no explicit failure ever raised (SHIELDA §3.1.30, 2025)
Boredom — *am I in a rut?*	**Einstellung effect / functional fixedness** — a known method blocks a better one (Luchins, 1942; Duncker, 1945)	**Reasoning cycles** — the loop re-queries information it already holds and re-derives conclusions it already reached, generating activity without forward progress (ReAct via SHIELDA §3.1.6, 2025)
Embarrassment / the rubber-duck moment — *can I actually explain this?*	**Illusion of explanatory depth** (Rozenblit & Keil, 2002); **confabulation** — fluent reasons for conclusions reached otherwise (Nisbett & Wilson, 1977)	**Logical soft errors** — covert reasoning degradation that lowers quality without ever throwing an exception (accumulated-semantic-ambiguity analyses, 2026)
The need to clear your head — *is my working context contaminated?*	**Anchoring**; cognitive overload from irrelevant priors (Tversky & Kahneman, 1974)	**Context rot / contamination** — uncurated accumulation produces noise, contradiction, and bloated state with no entropy sink (entropy-principle analyses, 2026)

One piece of corroboration is worth flagging, because it wasn't imported from the psychology side of the argument. In describing misaligned memory recall — where an agent retrieves entries similar to the current input but drawn from tasks with incompatible goals — the SHIELDA taxonomy notes that the resulting outputs are semantically disconnected while remaining fluent on the surface. That is this article's central claim, reached independently from inside the agent-reliability literature: fluency and correctness come apart, and the gap between them is where the failure hides. The isomorphism is doing less work here than it might appear; both fields arrived at the observation on their own.

Read top to bottom, the table says one thing eight times: **a closed loop drifts unless something outside it forces contact with reality, and in humans that something is largely emotion.** Feelings are not decoration on cognition or noise to be suppressed in favor of pure reason — they are the control layer that evolved precisely because a reasoning loop with no entropy sink is dangerous. Doubt is institutionalized as the demand to check; boredom as the demand to explore; frustration as the demand to change strategy; regret as the demand to cut losses. They are imperfect and miscalibrated in exactly the ways the "human failure" column records, but their *absence* is worse, which is what the "agent failure" column demonstrates.

The table has to survive its own doubt-governor, so name the risk plainly: a mapping this tidy invites suspicion, because *any* two rich failure taxonomies can be aligned if you are motivated to align them, and the rows are not equally tight — doubt → confirmation-loop and embarrassment → logical-soft-errors are near-identities, while boredom → Einstellung → perseveration is a three-step reach. The claim the table can honestly make is *structural* — same control function, same failure when it goes missing — not that every pairing is exact. What would promote it from a satisfying description to a load-bearing theory is prediction: does the isomorphism point to a governor, or an agent failure, that we did not already have a name for? Until it does, read it as a strong organizing analogy that is wearing its own yellow flag.

This reframes Section 7's prescription in the strongest possible terms. "Wire the read head to evidence" is not hygiene advice; it is the project of **hand-installing artificial governors** into a system that has none. A test suite is institutionalized doubt. A halt-on-failure condition is institutionalized frustration. A fresh context window is institutionalized *clear-your-head*. An external reviewer *can* be an entropy sink the loop cannot supply for itself — but only when the reviewer is themselves wired to reality; a reviewer who rubber-stamps the fluent diff is not a sink, just another node inside the loop. And it is the real argument for keeping a human in the loop — not as a supervisor by courtesy, but as the **governor of last resort** in an architecture that is otherwise a Turing-complete machine with the emotional regulation of a thermostat that has been told it is always the right temperature. The feelings this article catalogues are the governor set. The agent failure bestiary is what you get without them.

A note on evidentiary weight, since the table leans on it. SHIELDA is a systematic review of 55 studies plus one engineered case study, and its authors disclose that the AgentOps component was simulated through manual log analysis rather than implemented. Their own threats-to-validity section is candid about generalising from a single case. It is a strong source for *which failure modes have been named and observed* and a weak one for *how often they occur or whether the proposed fixes work* — and it is cited here only in the first sense.

---

9. Turning the feelings into a practice

The reframe is only useful if it changes what you do at the keyboard. The move in every case is the same shape: **do not act on the feeling; act on what the feeling is pointing at.**

When you notice…	The gauge is reporting…	The disciplined response
Pleasure at how clever this is	You optimized for intricacy, not correctness	Simplify until it's boring; ask if a colleague could debug it cold
"This is obviously right"	High FOR, possibly fluency-driven, not verification-driven	Find the one edge case that would prove it wrong; run that
A clean AI diff you want to just accept	Borrowed fluency masquerading as your own confidence	Read it line by line; run it; trace the actual output before merging
Doubt you keep talking yourself out of	A true low-evidence reading	Convert one claim to evidence — print, log, test, or read the source
"I basically understand this code"	Possible illusion of explanatory depth	Try to explain it line by line; the gaps are where the bugs are
Anxiety after you've already tested thoroughly	Unwarranted doubt (Costly quadrant)	Trust the evidence you gathered; stop re-checking the verified

Three principles unify the table. First, **fluency is the feeling of a claim; friction is the feeling of evidence** — and friction is the one to trust, because evidence is supposed to cost something. Second, the question that defuses almost every false signal is simply *what would I check if I doubted this?* — and then checking it, whether or not you currently feel the doubt. Third, obeying doubt has a cost, so it needs triage: not every claim is worth converting to evidence. Spend verification where the product of *how likely you are to be wrong* and *how much being wrong would cost* is high, and let genuinely low-stakes claims ride — a governor that fires on everything is just the Costly quadrant scaled up. "Go check" is the default the fluent world talks you out of; it is not a mandate to check all things equally.

Coda

Both emotions report the same underlying quantity: the **distance between your mental model and reality.** Cleverness is the feeling of admiring your model without testing it. Doubt is your model flagging that it has been running on untested inputs. Neither is the truth. Both are pointers at where the truth still needs to be checked. And they are only two of a larger governor set — boredom, frustration, regret, curiosity — that together form the control layer a reasoning loop cannot safely run without.

Working with a capable, fluent AI assistant raises the stakes of getting this right, because it makes the *sensation* of correctness cheaper than it has ever been while leaving the *fact* of correctness exactly as expensive as before. The buzz of "this is clever" and the chill of "I'm not sure" are the same instrument, reading the same gap, asking for the same response:

> **Go check.**

References

- Chromik, M., Eiband, M., Buchner, F., Krüger, A., & Butz, A. (2021). *I Think I Get Your Point, AI! The Illusion of Explanatory Depth in Explainable AI.* Proceedings of IUI '21.
- *DenoiseFlow: Uncertainty-Aware Denoising for Reliable LLM Agentic Workflows.* (2026). arXiv:2603.00532. [accumulated semantic ambiguity / logical soft errors]
- Duncker, K. (1945). *On problem-solving.* Psychological Monographs, 58(5). [functional fixedness]
- Feng, G., Zhang, B., Gu, Y., Ye, H., He, D., & Wang, L. (2023). *Towards Revealing the Mystery behind Chain of Thought: A Theoretical Perspective.* NeurIPS 2023.
- Fernbach, P. M., Rogers, T., Fox, C. R., & Sloman, S. A. (2013). *Political extremism is supported by an illusion of understanding.* Psychological Science, 24(6).
- Fisher, M., Goddu, M. K., & Keil, F. C. (2015). *Searching for explanations: How the Internet inflates estimates of internal knowledge.* Journal of Experimental Psychology: General, 144(3).
- Goddard, K., Roudsari, A., & Wyatt, J. C. (2012). *Automation bias: A systematic review of frequency, effect mediators, and mitigators.* JAMIA, 19(1).
- Hasher, L., Goldstein, D., & Toppino, T. (1977). *Frequency and the conference of referential validity.* Journal of Verbal Learning and Verbal Behavior, 16(1). [illusory-truth effect]
- International AI Safety Report (2026). §2.3.2, Risks to human autonomy — subsection "Automation bias persists with new AI tools."
- Kernighan, B. W., & Plauger, P. J. (1978). *The Elements of Programming Style* (2nd ed.). McGraw-Hill.
- Koch, A. S., & Forgas, J. P. (2012). *Feeling good and feeling truth: The interactive effects of mood and processing fluency on truth judgments.* Journal of Experimental Social Psychology, 48(2).
- Luchins, A. S. (1942). *Mechanization in problem solving: The effect of Einstellung.* Psychological Monographs, 54(6).
- Merrill, W., & Sabharwal, A. (2024). *The Expressive Power of Transformers with Chain of Thought.* ICLR 2024.
- Nisbett, R. E., & Wilson, T. D. (1977). *Telling more than we can know: Verbal reports on mental processes.* Psychological Review, 84(3). [confabulation]
- Nolen-Hoeksema, S. (1991). *Responses to depression and their effects on the duration of depressive episodes.* Journal of Abnormal Psychology, 100(4). [rumination]
- Perry, N., Srivastava, M., Kumar, D., & Boneh, D. (2022). *Do Users Write More Insecure Code with AI Assistants?* arXiv:2211.03622.
- Reber, R., & Schwarz, N. (1999). *Effects of perceptual fluency on judgments of truth.* Consciousness and Cognition, 8(3).
- Rozenblit, L., & Keil, F. (2002). *The misunderstood limits of folk science: An illusion of explanatory depth.* Cognitive Science, 26(5).
- Schuurmans, D. (2023). *Memory Augmented Large Language Models are Computationally Universal.* arXiv:2301.04589.
- Schuurmans, D., Dai, H., & Zanini, F. (2024). *Autoregressive Large Language Models are Computationally Universal.* arXiv:2410.03170.
- Schwarz, N. (2004). *Metacognitive experiences in consumer judgment and decision making.* Journal of Consumer Psychology, 14(4).

- *Silent Failure in LLM Agent Systems: The Entropy Principle.* (2026). arXiv:2606.08162. [no entropy sink / context rot]
- Staw, B. M. (1976). *Knee-deep in the big muddy: A study of escalating commitment to a chosen course of action.* Organizational Behavior and Human Performance, 16(1).
- Strathern, M. (1997). *'Improving ratings': audit in the British University system.* European Review, 5(3). [canonical phrasing of Goodhart's law]
- Thompson, V. A., Prowse Turner, J. A., & Pennycook, G. (2011). *Intuition, reason, and metacognition.* Cognitive Psychology, 63(3).
- Tversky, A., & Kahneman, D. (1974). *Judgment under uncertainty: Heuristics and biases.* Science, 185(4157). [anchoring]
- Unkelbach, C. (2007). *Reversing the truth effect: Learning the interpretation of processing fluency in judgments of truth.* Journal of Experimental Psychology: Learning, Memory, and Cognition, 33(1).
- Webster, D. M., & Kruglanski, A. W. (1994). *Individual differences in need for cognitive closure.* Journal of Personality and Social Psychology, 67(6). [premature closure]
- *Where LLM Agents Fail and How They Can Learn From Failures* (AgentErrorTaxonomy). (2025). arXiv:2509.25370. [error propagation / cascading failure]
- Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., et al. (2024). *OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments.* arXiv:2404.07972.
- Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). *ReAct: Synergizing Reasoning and Acting in Language Models.* ICLR 2023.
- Zhou, J., Chen, J., Lu, Q., Zhao, D., & Zhu, L. (2025). *SHIELDA: Structured Handling of Exceptions in LLM-Driven Agentic Workflows.* arXiv:2508.07935. [taxonomy of 36 exception types; §3.1.6 circular reasoning, §3.1.29 error propagation, §3.1.30 stopping too early, §3.1.11 misaligned memory recall]