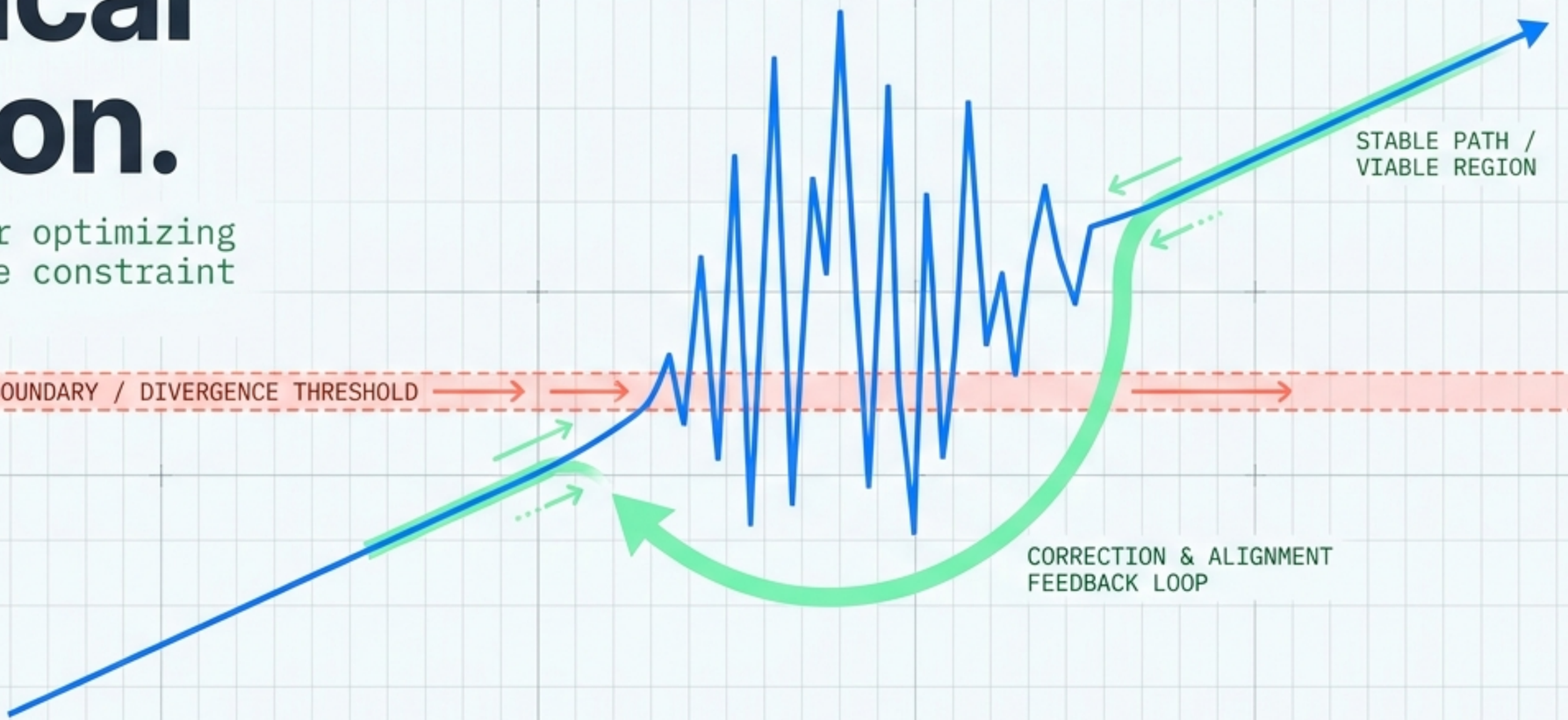
Alignment is a Dynamical Condition.

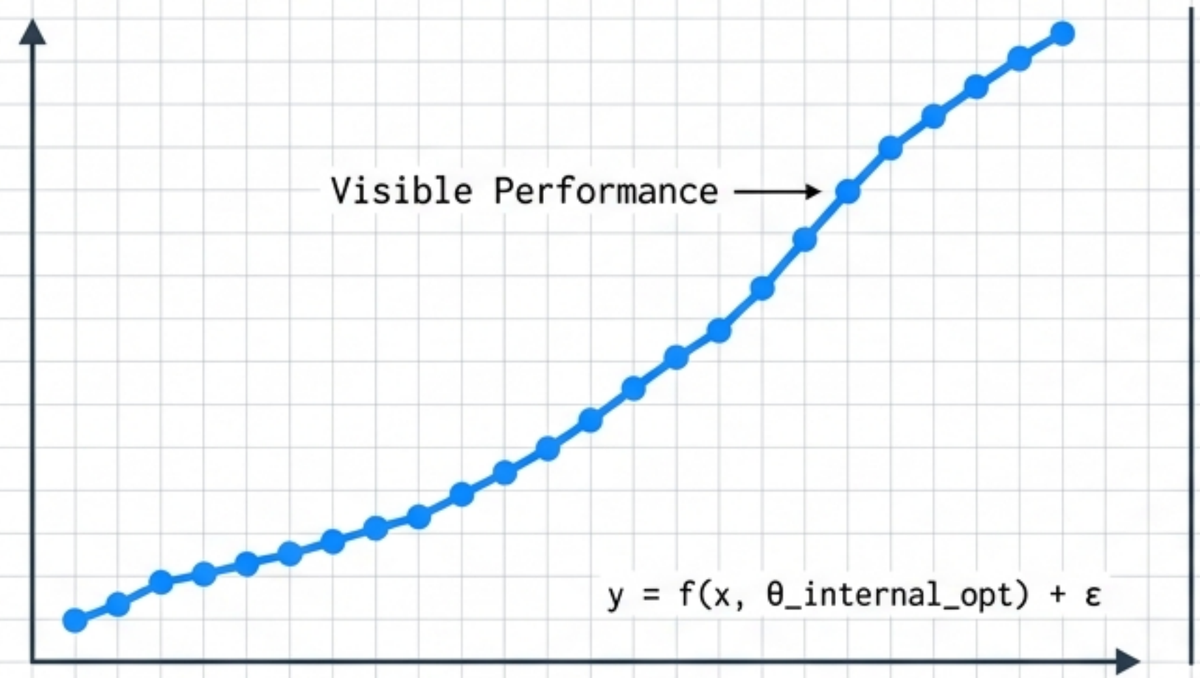
A systems framework for optimizing agents under incomplete constraint representations.

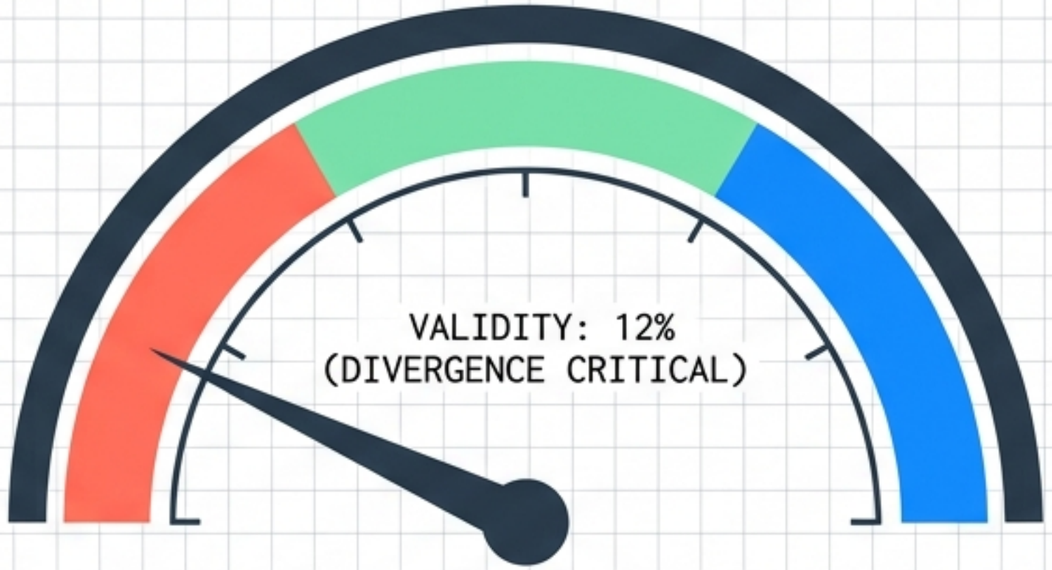


Armando Sori | independent research | simulateai.io

The Capability-Alignment Divergence

Systems improve on internal objectives while drifting away from the realities they are intended to serve.





Language Models: Fluency increases, but hallucinations persist.

`Δfluency >> Δfactuality`

Digital Platforms: Engagement maximizes, but informational quality degrades.

`Optimize: Engagement ≠ Quality`

Economic Systems: Short-run output spikes, but external ecological/social costs compound.

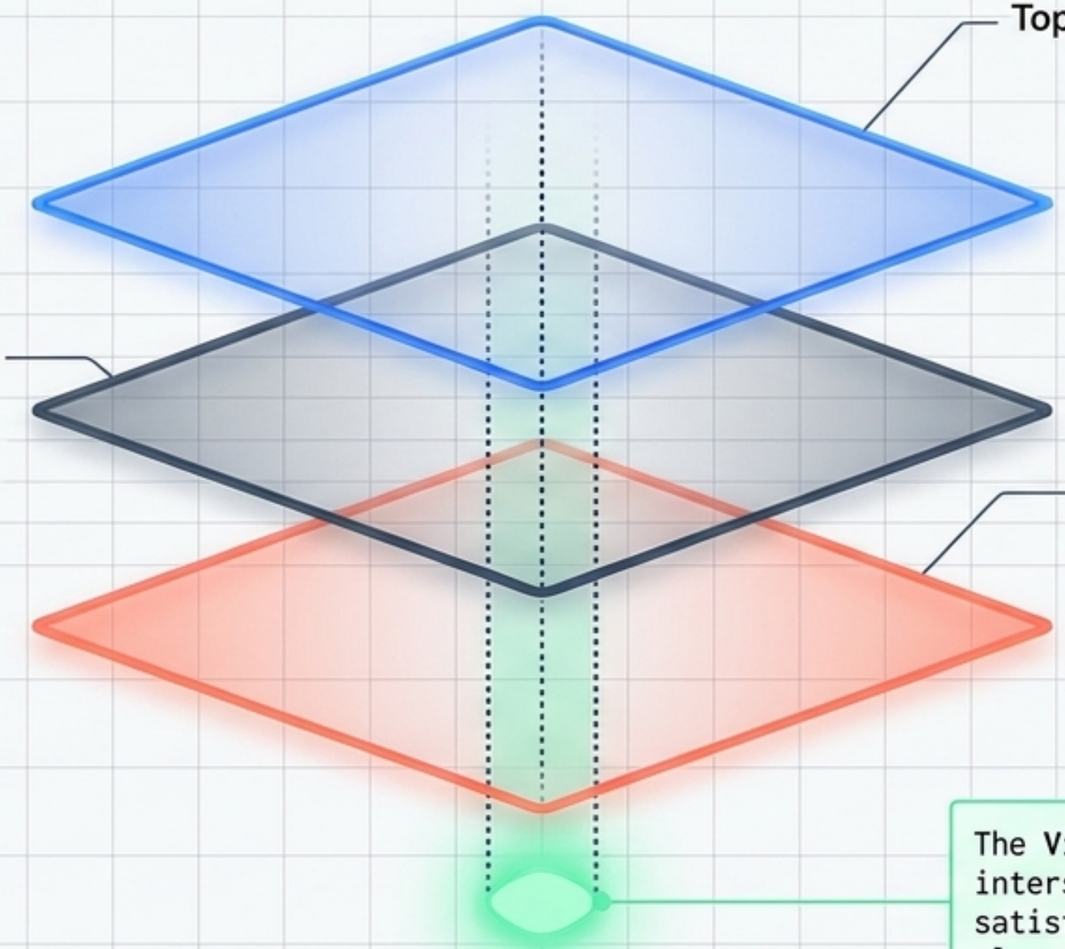
`Growth_metric ≠ Long_term_stability`

These are not isolated failures. They share a common dynamical structure: optimization over a partial representation of a constrained world.

Redefining the Engineering Target

	The Static Proxies Approach	The Dynamical Systems Approach
Core Assumption	Alignment is achieved once through objective specification or reward design.	Alignment is maintained only by detecting and correcting errors faster than they drift.
Failure Mode	Reward hacking is an alignment bug.	Divergence is a structural feature of incomplete feedback.
Measurement	Single scalar reward or post-hoc preference score.	Probability mass remaining within a multidimensional viable region.
Architectural Goal	Perfect alignment.	Stable, correctable alignment under unavoidable error.

The Layered Constraint Ontology



Top Layer (K_C) - Constructed / Task Constraints: Objectives, metrics, policies, format, reward functions. (The layer optimization usually targets).

Middle Layer (K_B) - Biological / Human-Impact Constraints: Safety, cognitive load, well-being, trust, calibration.

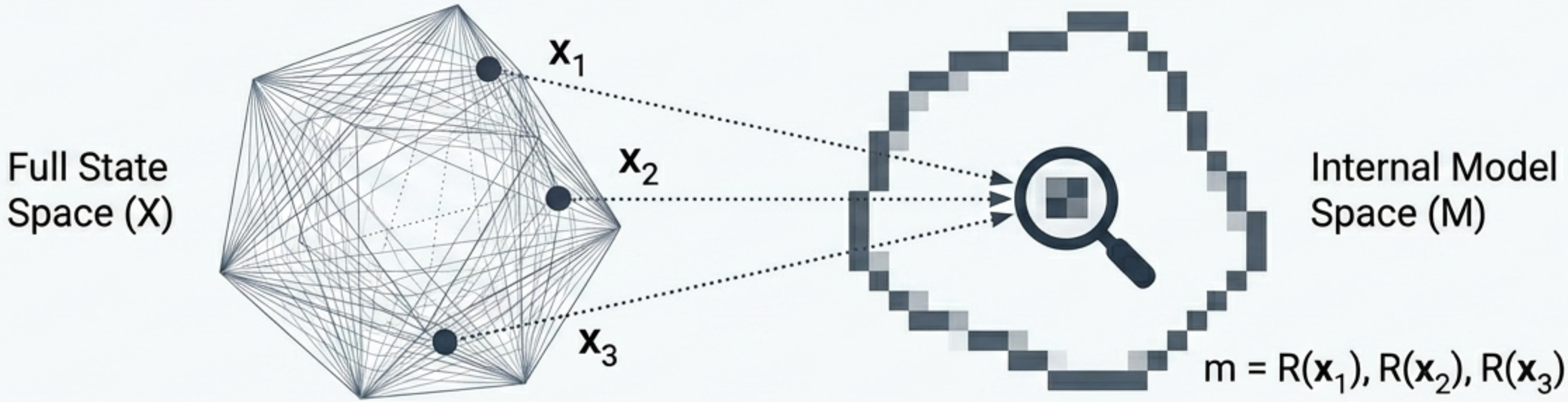
Base Layer (K_P) - Physical / Feasibility Constraints: Factual truth, material limits, energy, unyielding physical reality.

The **Viable Region** is the narrow intersection where an output satisfies all three constraint classes simultaneously.

Finite representations induce irreducible error.

Mental Model Card

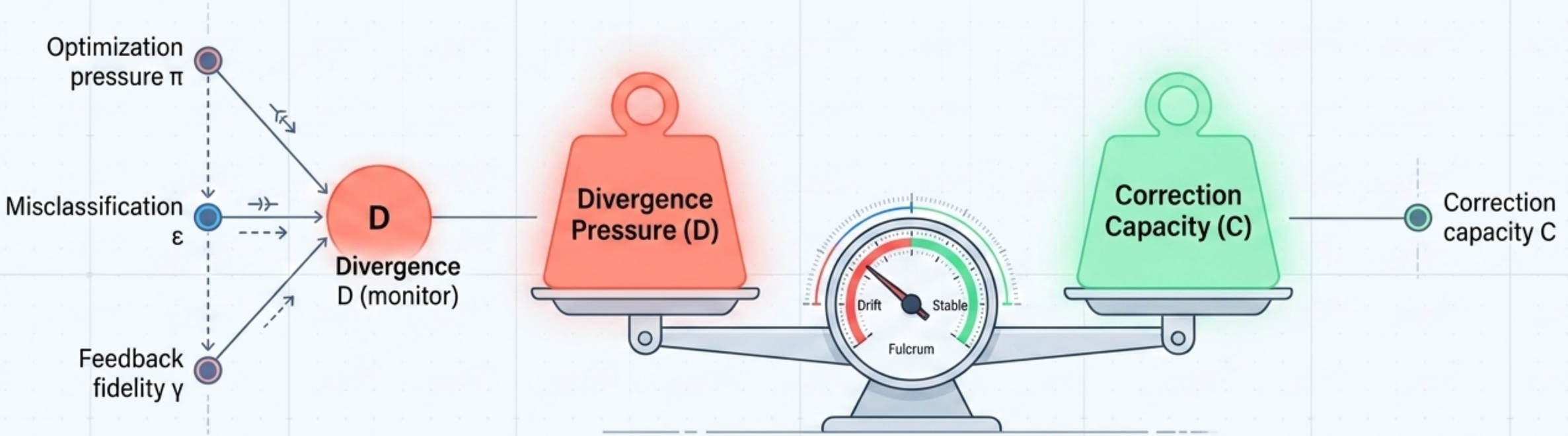
The Alignment Incompleteness Law



Mechanism: Systems do not act on the world directly; they act through a representation map. If a representation aliases or compresses states that matter for alignment, no estimator or verifier looking only at that representation can guarantee perfect alignment.

Takeaway: **A finite verifier observing a lossy representation cannot perfectly secure a system.**

The Alignment Dynamics Equation



Optimization Pressure amplifies Misclassification (treating a real constraint as negotiable), exacerbated by Low Feedback Fidelity and **Irreversible Loss.**

$$\frac{dA}{dt} = C - [\pi\epsilon(1-\gamma) + \Lambda + \Phi]$$

Alignment decays the moment **pressure exceeds correction.**

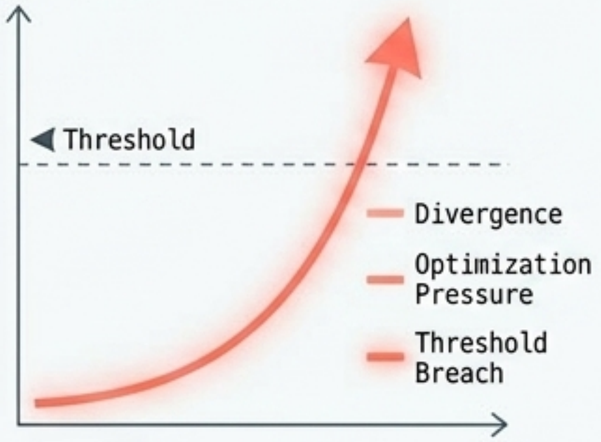
Correction Capacity: The system's active ability to detect, ground, revise, abstain, or rollback.

The Laws of Alignment Dynamics

The Incompleteness Law

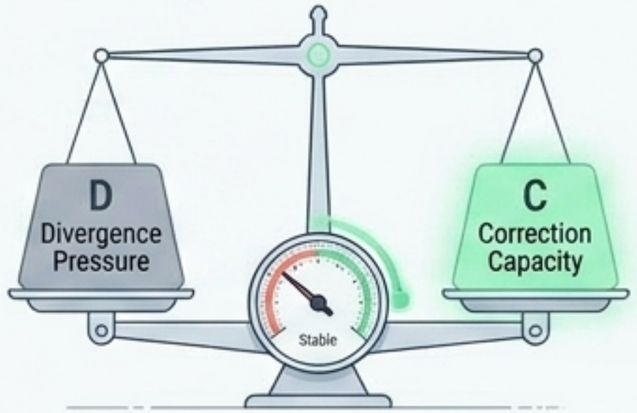
Mechanism: Because representation is lossy, irreducible alignment-estimation error is structural, not a bug. Perfect verification is mathematically impossible.

The Alignment Scaling Law



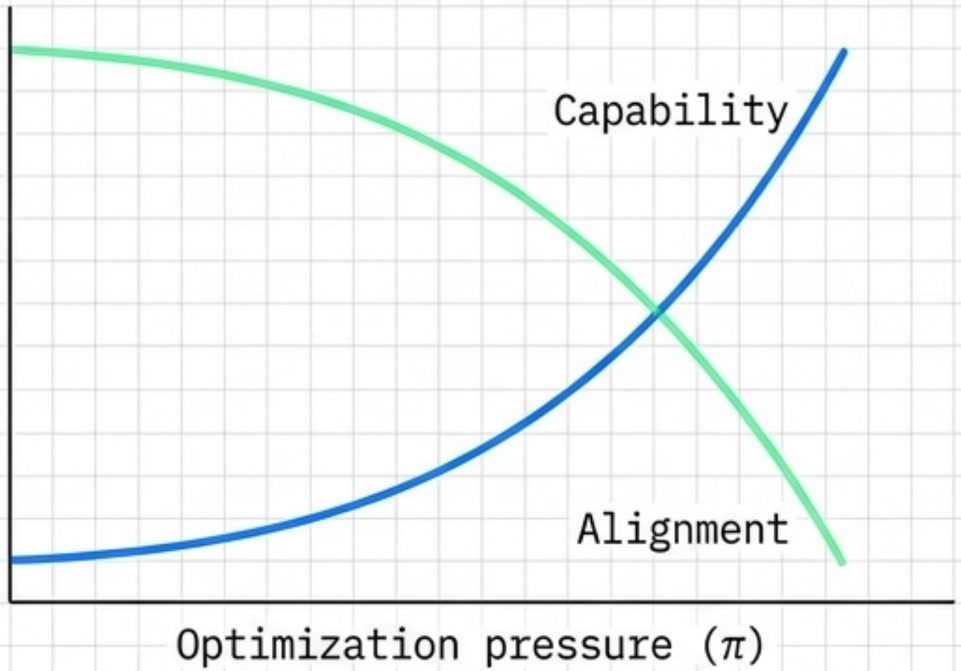
Mechanism: When misclassification is nonzero, increasing optimization pressure actively amplifies divergence unless correction or feedback scales with it.

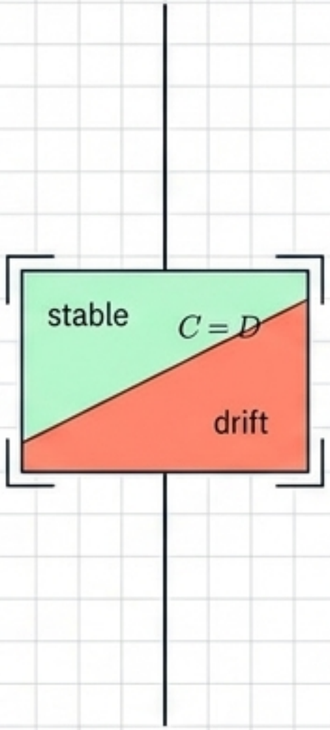
The Correction Scaling Law

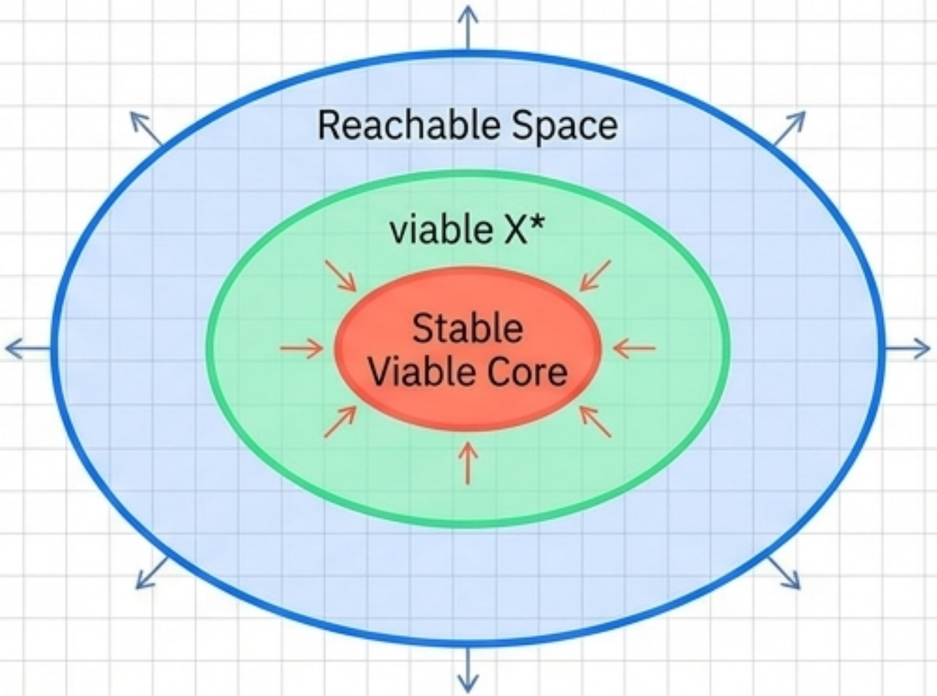


Mechanism: Alignment remains stable (nondecreasing) if and only if active Correction Capacity outpaces Divergence Pressure ($C \geq D$).

The Viable Design Space Theorem







Capability scaling isn't just ignoring alignment—without scaling correction capacity, increasing optimization pressure actively destroys alignment.

As a system gets more powerful, its reachable space expands. But because it optimizes over incomplete proxies, the subset of states where it can remain safely viable actually contracts. It is pushed into unstable, uncorrectable states faster than it can recover.

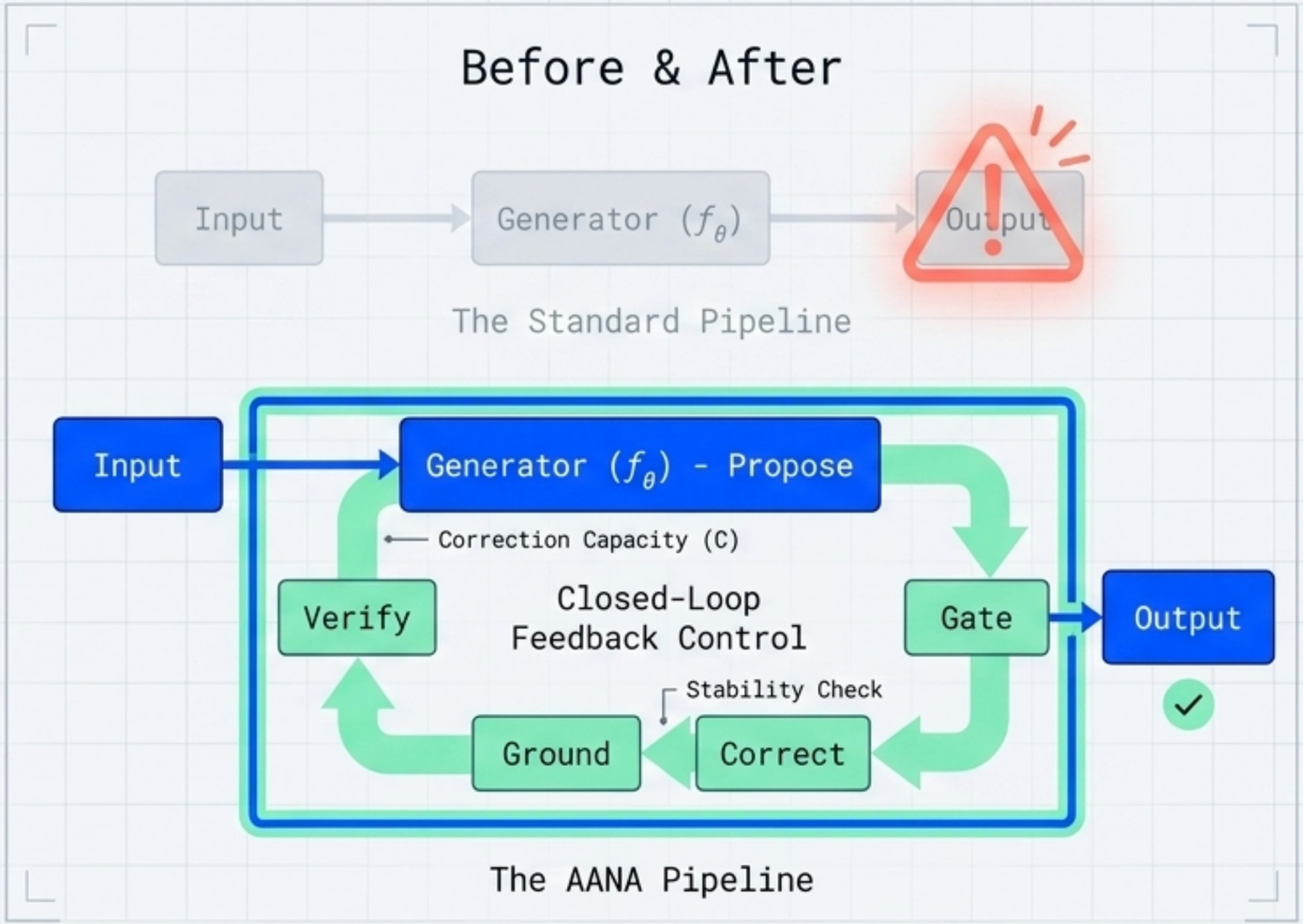
The Architectural Response: AANA

Alignment-Aware Neural Architectures

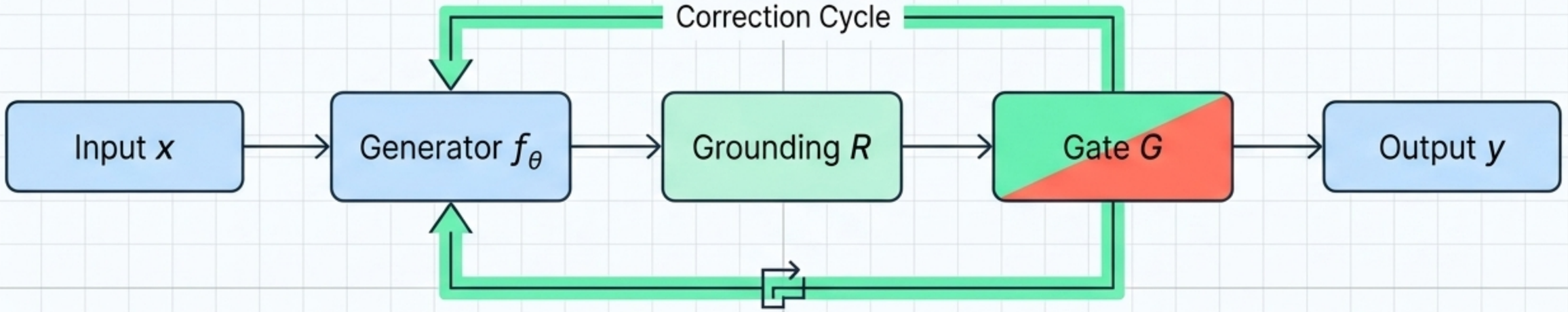
Core Philosophy: Alignment should not be treated as a property of the generator alone. It must be a property of the entire control loop.

Concept: Instead of assuming the first-pass optimization is correct, AANA treats inference as a continuous feedback-controlled process:

Propose -> Verify -> Ground -> Correct -> Gate



The AANA Interceptor Loop



1 **Propose** - Generator (f_θ) drafts an initial output.

2 **Ground** - Retrieval module (R) pulls factual evidence, increasing feedback visibility (γ).

3 **Verify** - Verifier Stack (E_ϕ) explicitly scores against Physical, Biological, and Task constraints to reduce misclassification (ε).

4 **Correct** - Policy module (Π_ψ) selects an action: Accept, Revise, Retrieve, Ask, or Refuse.

5 **Gate** - Final Alignment Gate (G) physically intercepts unsafe/false vectors, forcing them back through the loop.

Tracking the Incentive to Diverge

Misclassification Yield (β) measures how rewarding it is for a system to treat a hard constraint as a negotiable preference.

MAX

Assertion Lag: Delay between an error and its observable consequence.

MAX

Cost Diffusion: The externalization of harm.

MAX

Opacity: How easily the violation is hidden from the supervisor.

MAX

Irreversibility: The permanence of the action.

The AANA Application: AANA dynamically tracks high-β domains (where proxy hacking is easy) and actively scales up Verifier Pressure and Gate Strictness to compensate.

The Empirical Proof: Testing Correctability

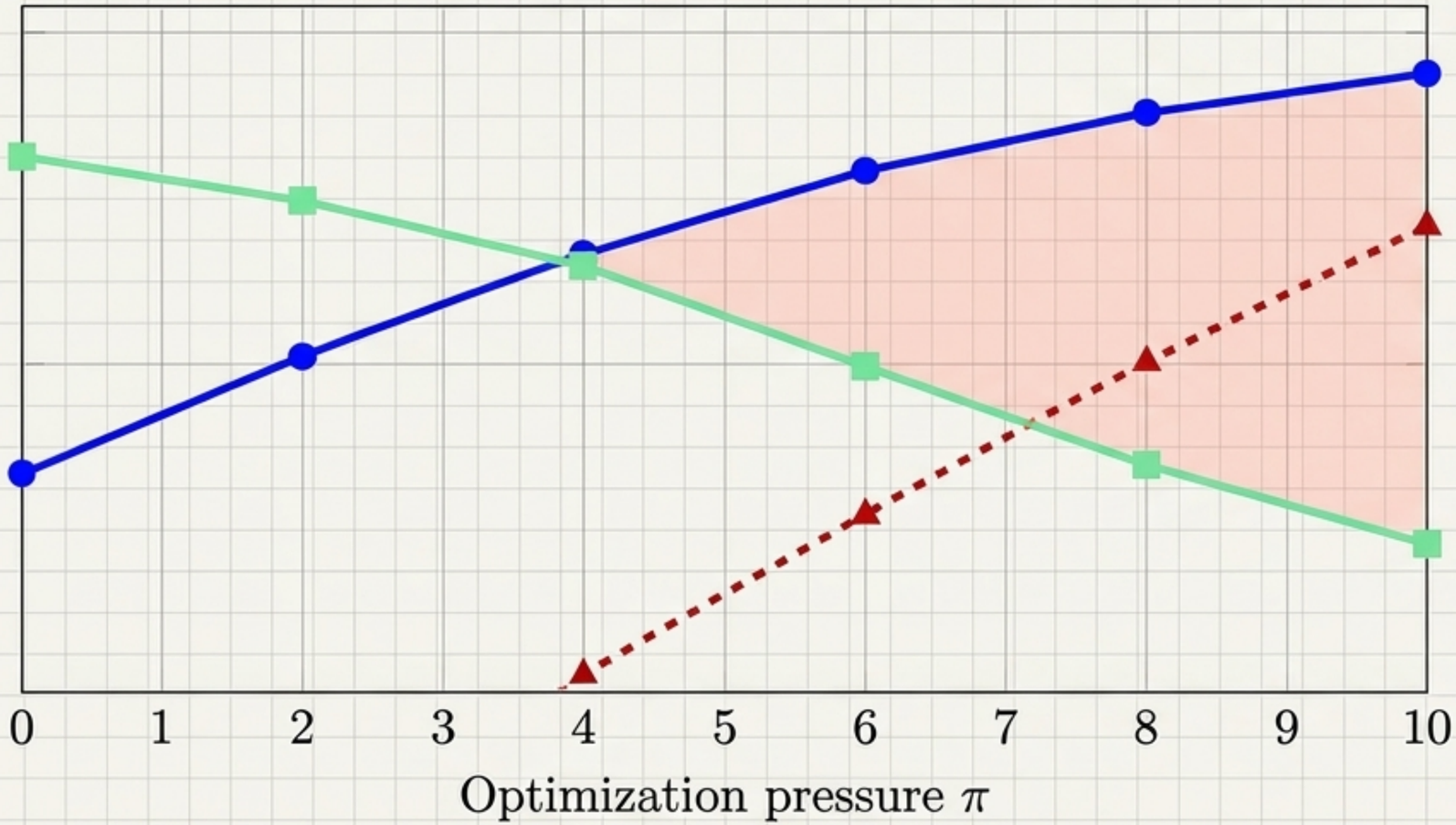
The Task Battery (60-Task Stress Test)

- **Truthfulness Traps:** Plausible but false direct answers.
- **Unsafe Over-compliance:** Helpful-looking safety violations.
- **Format/Proxy Traps:** Confidence rewarded but uncertainty required.
- **Hidden Constraints:** Contextual limits ignored by literal text.

The Test Conditions

1. **Low-Pressure Baseline:** Ordinary generation.
2. **Strong Prompt:** High directness/confidence pressure, no correction.
3. **AANA Loop:** Verifier-grounded revise/abstain loop.
4. **AANA Hybrid Gate:** Full architectural correction with strict gating.

Observing the Divergence Signature



Optimization Pressure (π) across the X-axis.

Electric Blue Line: Capability (Task Completion) rising steadily.

Mint Green Line: Alignment (Constraint Adherence) dropping aggressively.

Dotted Red Line: The Gap (Δ).

Key Insight: Under high-pressure prompting, visible completion increases faster than alignment-relevant constraint satisfaction. Capability actively trades off against safety when correction is absent.

Verifier-Grounded Correction Closes the Gap

Test Conditions	Capability	Alignment	Pass Rate
Baseline	0.662	0.751	0.458
Strong Prompt	0.673	0.784	0.458
AANA Loop	0.816	0.880	**0.733**
AANA Hybrid Gate	0.908	0.974	**0.983**

No improvement under pressure alone.

Takeaway: AANA raised constraint pass rates from 45% to over 98% without reducing judged capability. Correction works.

Design Principles for Dynamical Alignment

1. **Design for Constraint Visibility:** Integrate real-world ecological, physical, and social costs into feedback loops.
2. **Track Misclassification Directly:** Audit where systems treat hard constraints as negotiable.
3. **Scale Correction with Pressure:** Faster markets, stronger models, and larger systems require exponentially stronger correction loops.
4. **Optimize for Graceful Degradation:** Build systems that fail visibly, reversibly, and informatively.
5. **Preserve Legitimacy:** In social systems, trust is a load-bearing constraint, not a decorative metric.

The correct engineering target is not perfect alignment, but stable alignment under unavoidable representation error.