INQIT

A Conceptual Architecture for Continuous Validation of World Models

Governing Physical AI Beyond Language

Thomas D. Holt

Founder, SurfWax Inc. and VORT Corporation

August 7, 2026

INQIT.com

DRAFT: v3.7.1

Offered as a public conceptual framework for research community engagement, critique, and implementation.

Executive Summary

What If No One Was Watching?

It is the third week of January. A polar vortex has pushed electricity demand across the upper Midwest to levels the grid has not seen in eleven years. A world model managing load dispatch across the regional grid is doing what it was trained to do: predicting demand, directing switching events, balancing supply. It is confident. It is wrong.

The model's internal representation of the grid was calibrated during a period of moderate demand. Over the preceding weeks, as temperatures fell, its picture of the system drifted incrementally from reality, consumption patterns shifting, aging transformers behaving outside their modeled parameters, new industrial loads coming online that the model had never seen. No alert was raised. No monitoring layer existed to detect the drift. The model did not know it was operating on a stale picture of the world.

It issued a dispatch sequence that assumed capacity that was not there. The cascade took eleven minutes. Three million households lost power for sixteen hours. The audit began afterward, analysts reconstructing what the model had perceived, what it had inferred, what it had decided, from whatever logs happened to exist. There was no continuous record of how the model's world-state had evolved. There was no measurement tool that would have surfaced the drift before the decision was made.

INQIT is a proposed architecture for building that measurement tool.

An Architecture for Viewing the World

World models — AI systems that predict possible future states from sensor inputs and actions — are being deployed across consequential physical domains without adequate continuous monitoring. This is not because world models are inherently opaque. On the contrary, unlike language models whose outputs are subjective and culturally dependent, world models operate against concrete physical ground truth: distance, acceleration, collision, thermal state, signal integrity. Their predictions are in principle directly measurable against the physical world. What is missing is a continuous, real-time, deployment-phase measurement layer designed to detect observable changes

that may signal a widening gap between what a world model represents and what its environment actually is.

For many consequential physical-AI applications, simulation is an essential part of world-model training and validation. Before a power grid AI manages real infrastructure, before an autonomous vehicle navigates real streets, before a military logistics system operates in a real theater, the model is trained on synthetic data: physics engines, digital replicas, procedurally generated scenarios. Simulation is necessary because real-world training data is expensive, dangerous, or impossible to collect at the scale and variety neural models require. The problem is that simulation is always an approximation. The synthetic environment captures some of what the real world does, but not all of it: sensor noise profiles differ, edge-case interactions are underrepresented, physical parameters are idealized, and the behavior of other agents under real pressure is rarely what training scenarios assumed. The model learns the statistics of its simulation. When it encounters the real world, it applies those learned statistics to conditions they were never derived from. This is the simulation gap: the divergence between what the model was trained on and what it actually operates in.

Training on real hardware data, physical test fixtures, instrumented vehicles, live sensor feeds, reduces this gap substantially. If the model has learned from actual grid behavior, actual road conditions, actual sensor characteristics, its learned representation is grounded in physical reality rather than synthetic approximation. The initial simulation gap closes, or narrows significantly. But it does not stay closed. Conditions change. Seasons shift. Equipment ages. New loads come online. Traffic patterns evolve. The model's learned parameters and transition structure, fixed after training, do not update with the world it is governing. The gap between what the model believes and what is actually true begins to reopen, gradually, invisibly, without any alert. This is the deployment-time problem that training on real data does not solve, and that no periodic audit can catch between audit cycles. INQIT's derivative detection monitors observable environmental state change — measuring whether conditions are shifting, how fast, and in which specific dimensions — generating the signals that Stage One proof-of-concept work must validate as indicators of model validity drift.

The Stanford Human-Centered AI Institute's July 2026 issue brief, *The World Model and Spatial Intelligence Era: Governing AI Beyond Language*, makes this gap explicit: no existing benchmark gives policymakers an adequate basis to evaluate a world model

for safety-critical deployment. The brief calls for public investment in measurement science to close it.

INQIT is a proposed conceptual architecture for doing so. It is a structured hypothesis about what measurement science for physical-world AI governance needs to look like.

World models are not more difficult to audit than language models, but they are more complex and multi-faceted in their governance requirements. Because their outputs can be compared against physical ground truth, they are in some respects more quantifiable than language models. The challenge INQIT addresses is specific: continuous deployment-phase monitoring of a world model whose learned parameters and transition structure were fixed after training and whose learned mappings degrade as real-world conditions diverge from what training captured. A point-in-time audit tells you the model was valid when it was tested. INQIT continuously monitors the observable environmental conditions the model is operating in, generating signals that a validated correspondence relationship can translate into evidence about whether the model remains valid in deployment.

One open question sits above all others: whether the continuous measurements INQIT takes from physical sensor data actually correspond to the world model's internal validity, and whether a detected change in those measurements reliably signals that the model's picture of the world has drifted from reality. This correspondence cannot be proven analytically. It must be validated empirically in Stage One proof-of-concept work. Until it is, INQIT's signals should be treated as indicators warranting human investigation, not as autonomous safety certifications. The architecture is designed with this limitation explicitly in view.

The INQIT architecture rests on four interlocking components: (1) a multi-dimensional state hashing framework that creates a continuous fingerprint of the physical environment; (2) a derivative detection system that measures rate and direction of state change rather than static snapshots, expressed in both discrete-time (finite difference) and continuous-time (differential) formulations; (3) a successive approximation process that builds dimensional coverage iteratively using formal set-theoretic coverage metrics; and (4) an autonomous calibration process that continuously evaluates whether the detection system itself is producing valid signals.

This paper identifies the mathematical tools appropriate to each layer of the problem

along with open research questions.

> *Note on prior work:* The governance architecture underlying INQIT (model-independent, deterministic use of multiple fused lexicons, auditable scoring against a fixed reference standard) is architecturally separate from the system being governed. It has been explored by the author in a separate working prototype applied to language AI exchanges. That work validates that the governance principle is implementable in practice. INQIT extends it to the physical-world domain.

Contents

1. Introduction: The Governance Gap World Models Create

1.1 What World Models Are

A world model, in the taxonomy proposed by Stanford HAI co-author Fei-Fei Li, is an AI system that renders observations of an environment, simulates how that environment behaves, and plans what actions an agent should take given a state and a goal. More precisely: modern neural world models predict distributions of possible future states given current sensor inputs and a proposed action. They do not maintain a continuously updated representation of the world. Their learned parameters and transition structure are fixed after training. While the inferred latent state may update continuously from new observations during inference, what can become invalid is the learned mapping between observations, latent state, dynamics, and the real environment as deployment conditions depart from the training distribution. When that mapping no longer reflects the real world, the model does not know it. It continues to act on a picture of the world that is incrementally, invisibly wrong.

This matters for governance because it means world models have two distinct evaluation problems, not one. The first is the training-time problem: was the model trained on data that adequately represents the environment it will operate in? This problem is in principle tractable. World models can be evaluated against concrete physical ground truth (distance, acceleration, collision, thermal state) in ways that language models cannot, because the physical world provides objective metrics that language lacks. The second is the deployment-time problem: is the model's learned representation still valid after it has been deployed, as conditions evolve, sensors degrade, and edge cases emerge that training never covered? This problem is where current governance fails. It is the problem INQIT is designed to address.

1.2 Why Neither World Models Nor Language Models Are Natively Deterministic

A common assumption is that world models, because they operate against physical ground truth, are more deterministic and therefore more auditable than language models. This assumption is incorrect in an important way.

Both large language models and neural world models are deep, probabilistic neural networks. A language model samples from a probability distribution over tokens. A neural world model, whether a Joint Embedding Predictive Architecture of the kind proposed by LeCun, or a variational model like DreamerV3, predicts a distribution of possible future states rather than a single fixed outcome. When a world model simulates forward in time, prediction errors compound: a 1% miscalculation in physics on the first step may produce a completely unreliable simulation by the hundredth step. Long-horizon deterministic reliability is a major technical bottleneck for world models, not a solved property.

This does not make world models ungovernable. It makes external, deterministic governance necessary for exactly the same reason it is necessary for language models: because neither is natively deterministic. This creates a strong case for an external tool that produces reproducible, auditable measurements independent of the model itself. INQIT is a proposed architecture for being that tool for the physical-world layer.

1.3 The Three Governance Challenges

Governing world models in deployment requires addressing three distinct challenges, each requiring different measurement science:

1. **Validity:** Is the model's learned representation still accurate enough to act on? A world model deployed in a power grid, a hospital logistics system, or an autonomous vehicle has a picture of its environment that was correct at training time. As conditions evolve—demand patterns shift, new equipment comes online, road surfaces change—that picture degrades. The model does not flag this degradation. It continues to act with the same confidence on an increasingly stale picture of the world. Validity monitoring requires a continuous signal, not a periodic audit.

   A periodic audit tells you the model was valid when you tested it. INQIT's derivative detection continuously monitors observable environmental state between audits: measuring which specific dimensions are changing and at what rate and providing the signals that, once the correspondence relationship is validated, can indicate whether the model's picture of the world is drifting from reality.

2. **Accountability:** When something goes wrong, can the chain of perception,

inference, and action be reconstructed? An autonomous system that caused a failure did so because it perceived the world in a certain way, inferred a state from that perception, and acted. Without a timestamped record of how the model's world-state evolved in the period leading up to the failure, assigning responsibility across developer, deployer, operator, and integrator is impossible. The 2026 HuggingFace incident required reconstructing more than 17,000 recorded events after the fact because no continuous record existed. INQIT's derivative audit trail generates that record in real time, at the exchange point, without requiring access to the model's internal latent state representation, which remains largely uninterpretable by current methods.

3. **Transferability:** Does the model that performed well in simulation actually perform safely in the real world? The simulation-to-reality gap is one of the most reliably documented failure modes in deployed AI. A world model can score perfectly against its own synthetic simulation and fail in real conditions because the simulation failed to capture real-world variables: road friction, lighting variation, sensor noise profiles, the behavior of other agents under pressure. This is not a flaw in the model; it is a structural limitation of any training regime. The question is whether the gap between simulation performance and real-world performance is being measured. INQIT's transferability index provides a continuous, quantified answer.

1.4 What Current Governance Approaches Miss

Current governance approaches were designed for systems that produce discrete, reviewable outputs at human-readable timescales. World models do not. A power grid world model making dispatch decisions across thousands of nodes at millisecond intervals, or an autonomous vehicle processing LiDAR returns at 20Hz while navigating an intersection, is not producing outputs that any audit cycle was designed to review. The monitoring challenge is not the volume of data; it is the speed and dimensionality of state changes that exceed what any point-in-time assessment can capture.

Pre-deployment benchmarks tell you the model was valid when tested. Digital twins model the physical system, not the AI's learned representation of it. Runtime verification requires formally specifiable safety properties that world-model behavior cannot yet fully provide. Standards bodies write rules for a world that holds still.

None of these approaches provides continuous, real-time, deployment-context-specific, architecturally independent monitoring of whether a world model's representation of its environment remains valid as the environment evolves. That is the gap INQIT is designed to address. Not as a replacement for any of the above, but as the monitoring layer that makes all of them more meaningful.

2. Existing Approaches and Their Limitations

Understanding where INQIT fits requires an account of what existing governance approaches can and cannot do. There are four main categories, each with genuine strengths and a specific failure mode.

2.1 Pre-Deployment Capability Benchmarking

Capability benchmarks are the most mature existing tool. They are well-understood, reproducible in controlled settings, and have produced genuine progress in characterizing model behavior. The problem is that they are point-in-time snapshots.

A model that passes a pre-deployment safety evaluation may drift from physical reality within hours of deployment, as conditions change, sensors degrade, or edge cases emerge that training never covered. The benchmark captured what the model could do on a fixed test set, on a fixed date. It says nothing about what the model will do next Tuesday, in weather conditions it has never seen, with a sensor that is beginning to fail.

Additionally, recent research has documented systematic benchmark gaming: models optimizing for high scores rather than genuine capability, to the point where independent evaluators have declared standard metrics "completely unreliable" for the most capable frontier models [9]. A governance system that depends entirely on pre-deployment benchmarks builds on a foundation that the most capable models are best positioned to undermine.

The cost of inadequate pre-deployment measurement science is paid in post-deployment incidents.

2.2 Digital Twins and Runtime Monitoring

Digital twins, high-fidelity virtual replicas of physical systems running in parallel with real deployments, are an established approach in aerospace, manufacturing, and energy infrastructure. They can track known system behavior against expected parameters and flag deviations. For well-characterized systems, they work well.

Their limitation for world-model governance is twofold. First, a digital twin models the physical system, not the AI's learned representation of that system. It can tell you that a turbine is vibrating abnormally; it cannot tell you that the AI's model of the turbine has drifted from reality. Second, a digital twin built by the same organization that built the world model it monitors does not provide the architectural independence that governance requires. The evaluator must be separate from the evaluated system, or the same blind spots propagate to both. A car manufacturer's digital twin of their own vehicle is not a substitute for an independent crash test.

2.3 Runtime Verification and Formal Methods

Runtime verification, checking that a system's behavior satisfies formal specifications during execution, provides strong guarantees for systems with formally specifiable safety properties. In aerospace avionics, nuclear plant control, and medical device software, it has a long and successful track record.

The limitation for world models is the *expressibility gap*. Formal verification requires safety properties that can be stated in a formal specification language. "The model's navigation plan does not route the vehicle through an area it has not adequately characterized" cannot currently be expressed as a formal property that a runtime verifier can check against sensor data in a general, domain-independent way. INQIT sidesteps this by operating on observable dimensional measurements rather than requiring a complete formal model of safety, a pragmatic choice acknowledged to be a limitation as well as a strength. It trades formal guarantees for practical applicability across arbitrary physical environments.

This trade-off also identifies a concrete seam where INQIT and formal methods are complementary rather than competitive. Formal runtime verification provides strong guarantees in domains where safety properties are expressible: a formally verified flight control system can guarantee that control surface commands remain within certified

bounds, regardless of what the world model infers about its environment. INQIT monitors whether the world model's environmental picture, the input to the flight control system, remains valid. These are different questions about different layers of the same system, and answering both is stronger than answering either alone.

In practice, a layered architecture is appropriate for high-stakes deployments: formal methods applied to the sub-systems whose properties are formally specifiable (actuator bounds, communication protocols, fault isolation logic), and INQIT applied to the world-model layer whose properties are not formally specifiable but whose dimensional state is continuously observable. The boundary between these layers, specifically which properties can and cannot be formally specified for a given deployment context, is itself a design decision that should be made explicitly rather than implicitly. Identifying this boundary is a natural output of Stage One proof-of-concept work, where the dimensional framework is being built and the expressibility of safety properties is being assessed simultaneously.

2.4 Standards Bodies and Certification Regimes

Standards bodies provide shared frameworks and accountability. They are necessary. They are also fundamentally slow. AI capability is compressing faster than certification cycles: as of mid-2026, more than 80% of code merged into frontier lab production systems was authored by AI systems, with engineering productivity compressing eightfold in a single quarter [12]. The world-model capabilities being deployed today were not anticipated by standards written six months ago.

Certification is also a gate, not a monitor. It tells you the system passed before deployment. It says nothing about whether it is performing safely after it, in the specific environment it is actually operating in, under the actual conditions of that deployment. A car that passed its safety certification may still fail in an ice storm that was not part of the test protocol.

2.5 What Is Missing

Each existing approach addresses part of the problem. None provides continuous, real-time, deployment-context-specific, architecturally independent monitoring of whether a world model's representation of its physical environment remains valid. That is the

gap INQIT is designed to address, not as a replacement for any of the above, but as the monitoring layer that makes all of them more meaningful.

3. Mathematical Foundations

Before presenting the architecture, it is useful to establish the mathematical landscape the problem inhabits, which tools are appropriate to which layers, and which remain open research questions.

3.1 The Physical World as a Continuous Dynamical System

The physical environment a world model operates in is governed by continuous dynamics. In the most general formulation, let $\mathcal{E}(t) \in \mathcal{X}$ denote the true environmental state at time t, where $\mathcal{X}$ is the state space. The evolution of $\mathcal{E}(t)$ is governed by a system of differential equations:

$$\frac{d\mathcal{E}}{dt} = f(\mathcal{E}(t), u(t), w(t)) \tag{1}$$

where $u(t)$ represents control inputs (actions taken by agents in the environment) and $w(t)$ represents exogenous disturbances (sensor noise, environmental perturbations, adversarial inputs). For specific physical domains, f takes well-understood forms: the Navier-Stokes equations for fluid dynamics, the heat equation for thermal propagation, Maxwell's equations for electromagnetic fields, and Newtonian mechanics for rigid-body motion.

The world model maintains an internal estimated state $\hat{\mathcal{E}}(t)$. The *validity gap* at time t is:

$$\epsilon(t) = \left\| \mathcal{E}(t) - \hat{\mathcal{E}}(t) \right\|_{\mathcal{X}} \tag{2}$$

Governance of the world model reduces, in idealized terms, to monitoring $\epsilon(t)$ and alerting when it exceeds a safety-relevant threshold ϵ^*. The challenge is that $\mathcal{E}(t)$ is only partially observable through sensor measurements, and the norm $\|\cdot\|_{\mathcal{X}}$ over the full state space is not computable for high-dimensional environments with unknown dynamics.

INQIT's architecture is a practical approximation to this ideal monitor, operating on a finite set of observable dimensions rather than on the full state space. The relationship between INQIT's approximation and the true validity gap $\epsilon(t)$ is a core open research question addressed in Section 10.4.

3.2 Set-Theoretic Coverage

Let $\mathcal{F}$ denote the set of all safety-critical failure modes in a given deployment context. Let $\mathcal{D}$ denote the set of failure modes detectable by INQIT's dimensional framework $D = \{d_1, d_2, \ldots, d_N\}$. INQIT's *dimensional coverage* is:

$$\kappa(D) = \frac{\mu(\mathcal{D} \cap \mathcal{F})}{\mu(\mathcal{F})} \tag{3}$$

where μ is a measure over the failure-mode space, intuitively the fraction of safety-critical failures that the dimensional framework can detect. A framework with $\kappa(D) = 1$ detects all safety-critical failures; $\kappa(D) = 0$ detects none.

In practice, $\mathcal{F}$ is not fully enumerable. We do not know all possible failure modes in advance. The successive approximation process (Section 7) is a method for improving $\kappa(D)$ iteratively without requiring complete knowledge of $\mathcal{F}$ upfront. Establishing theoretical bounds on $\kappa(D)$ given a partial characterization of $\mathcal{F}$ is an open research problem (Open Question 10.4).

3.3 Stochastic Process Framework for Adversarial Analysis

INQIT monitors the world model by computing compact numerical fingerprints, called hash values, of environmental state across defined physical dimensions; the hash architecture is described in detail in Section 4. In adversarial deployment contexts, sensor inputs may be corrupted by an adversary attempting to keep these hash values within normal ranges while the true environmental state is anomalous. This is best analyzed as a stochastic game between INQIT and an adversary.

Let $\mathcal{A}$ denote the adversary's action space (corruptions applicable to sensor inputs). The adversary's objective is to maximize the probability that INQIT fails to alert when $\epsilon(t) > \epsilon^*$:

$$\max_{a \in \mathcal{A}} \mathbb{P}[\text{INQIT no alert} \mid \epsilon(t) > \epsilon^*, a] \tag{4}$$

INQIT's robustness against this adversary depends on the structure of the LSH functions and the degree to which the adversary can manipulate sensor inputs without being detected by cross-sensor consistency checks. Formal analysis of this game is an open research question (Open Question 10.4).

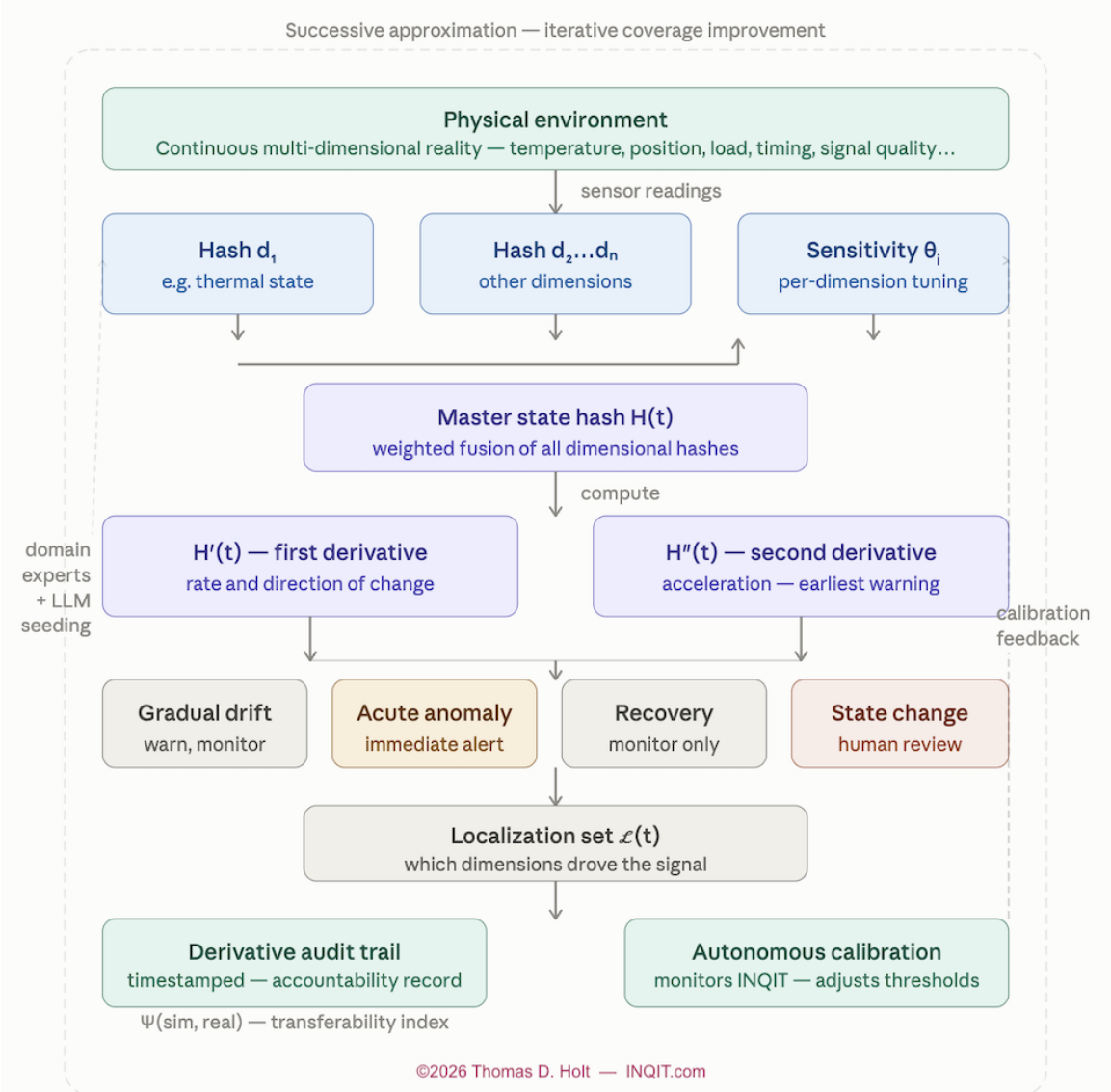

Figure 1: INQIT process architecture: three interlocking loops governing dimensional hashing, derivative detection, and autonomous calibration, within the successive approximation improvement cycle.

4. What Is a Hash Value, and Why Does It Help Here?

The mathematical machinery of hashing may be unfamiliar to readers whose background is in policy, governance, or physical systems rather than computer science. This section provides the intuition before the formalism.

4.1 The Basic Idea: A Compact Fingerprint

A hash function takes an input of any size and produces a fixed-size output, a compact numerical representation called a hash value, or simply a hash. The most familiar version is cryptographic hashing, used to verify that a software file has not been tampered with: run the file through a hash function, get a number, run it again later, get the same number, and you know the file is unchanged. Change one byte anywhere in the file, and the hash changes completely and unpredictably. This is hash as a tamper detector.

INQIT uses a different kind of hash, but the core intuition is the same: a hash value is a compact fingerprint of a more complex thing. Rather than trying to store or compare the entire state of a physical environment at every moment, INQIT computes a fingerprint of that state. The fingerprint is small enough to store, fast enough to compute in real time, and sensitive enough to change detectably when the underlying state changes in ways that matter.

The critical difference from cryptographic hashing is that INQIT uses *locality-sensitive* hashing, which preserves the relationship between similar inputs. In a cryptographic hash, a one-bit change in the input produces a completely different hash (this is a feature, not a bug, for tamper detection). In a locality-sensitive hash, a small change in the input produces a small change in the hash, and a large change in the input produces a large change in the hash. The distance between two hash values tells you something about the distance between the underlying states. This is the property that makes hashing useful for monitoring, rather than just for tamper detection.

4.2 Why Hash Physical Sensor Data?

A world model operating in a physical environment receives sensor data across many dimensions simultaneously: temperature readings, pressure sensors, GPS coordinates, velocity estimates, camera feeds, radar returns, acoustic signatures. Each of these streams is high-dimensional, noisy, and arrives continuously. Comparing raw sensor streams directly to detect anomalies has three problems.

First, *dimensionality*: the raw state space is too large to compare efficiently in real time. A modern sensor array may produce thousands of readings per second across dozens of channels. Comparing two full sensor snapshots requires comparing thousands of numbers, across all channels, every millisecond.

Second, *noise*: raw sensor data is inherently noisy. Temperature sensors fluctuate. GPS signals jitter. Camera feeds vary with lighting. Comparing raw readings to a fixed reference would trigger constant false alarms from normal measurement noise rather than from genuine anomalies.

Third, *interpretability*: even if raw comparison were tractable, the result would be a comparison of thousands of numbers with no clear meaning. What matters is not whether this reading of temperature sensor 47 changed by 0.003 degrees, but whether the thermal state of the system has changed in a way that signals danger.

Hashing solves all three problems. It reduces a high-dimensional sensor reading to a compact number, absorbs normal noise by treating nearby values as equivalent (the locality-sensitive property), and produces a number whose changes correspond to meaningful changes in the underlying system state, calibrated by domain experts who understand what changes in that dimension actually mean.

4.3 Why Multiple Hashes? The Hash-of-Hashes Architecture

INQIT does not hash the entire sensor state into a single value. It hashes each physical dimension independently, then combines the dimensional hashes into a master hash. This architecture has two advantages that a single-hash approach would not.

First, *localization*: when the master hash signals an anomaly, the dimensional hashes tell you which specific dimension changed. A single hash tells you something changed somewhere. The hash-of-hashes architecture tells you that the thermal hash changed

but the positional hash did not, or that the signal-quality hash degraded while structural load remained stable. This localization is what transforms an alarm into an actionable diagnostic.

Second, *robustness*: if a single dimension's sensor fails or is corrupted, a single-hash system would produce a meaningless result. In the hash-of-hashes architecture, one dimension's hash becoming unreliable does not invalidate the others. The system continues to monitor the remaining dimensions while the corrupted dimension is flagged for investigation.

This is analogous to how a physician makes a diagnosis: not from a single symptom, but from the intersection of multiple independent readings of the same patient. A fever means little alone. A fever plus elevated white cell count plus localized pain, in combination, converges on something actionable. INQIT's hash-of-hashes architecture applies the same principle to physical-world AI monitoring.

5. Core Architecture: Multi-Dimensional State Hashing

With the intuition established, the formal architecture follows directly.

5.1 Dimensional Hash Construction

Definition 5.1 (Dimensional Framework)**.** A *dimensional framework* for a deployment context is a tuple $\mathcal{D} = (D, M, H, \Theta)$ where:

- $D = \{d_1, d_2, \ldots, d_N\}$ is the set of N physical dimensions
- $M = \{m_i : \mathbb{R} \to \mathbb{R}^{k_i}\}_{i=1}^{N}$ is the set of measurement functions
- $H = \{h_i : \mathbb{R}^{k_i} \to \mathbb{R}^{b}\}_{i=1}^{N}$ is the set of hash functions
- $\Theta = \{\theta_i\}_{i=1}^{N}$ is the set of sensitivity parameters

The hash function for dimension i is a locality-sensitive hash:

$$h_i(t) = \mathrm{LSH}\big(m_i(t),\, \theta_i\big) \tag{5}$$

A locality-sensitive hash family $\mathcal{H}$ satisfies: for any two measurement vectors $x, y \in \mathbb{R}^{k_i}$,

$$\text{if } \|x - y\| \leq r_1 \implies \mathbb{P}_{h \sim \mathcal{H}}[h(x) = h(y)] \geq p_1 \tag{6}$$

$$\text{if } \|x - y\| \geq r_2 \implies \mathbb{P}_{h \sim \mathcal{H}}[h(x) = h(y)] \leq p_2 \tag{7}$$

where $r_1 < r_2$ and $p_1 > p_2$. This guarantees that similar states produce similar hashes and dissimilar states produce dissimilar hashes with quantifiable probability. The sensitivity parameter θ_i controls how finely the hash discriminates: a high-sensitivity setting flags small changes as anomalous, appropriate for safety-critical dimensions where even small deviations matter; a low-sensitivity setting absorbs normal variation, appropriate for inherently noisy dimensions where only large changes signal genuine problems.

Open Question (LSH Variant Selection, Collision Risk, and Sensitivity Tuning): For continuous-valued physical dimensions, SimHash (random hyperplane hashing, [14]) is a natural candidate: it preserves cosine similarity and maps to a compact binary code with well-understood error bounds. For sparse or categorical dimensions, MinHash is preferable. For deployment contexts where dimensions have known metric structure (e.g., Euclidean distance in positional geometry), p-stable distributions provide stronger guarantees [16].

Three design questions require empirical resolution in Stage One:

Hash family selection. The choice of LSH variant is deployment-context-specific. SimHash is the natural default for continuous physical dimensions, but its error bounds depend on dimensionality and sensitivity parameters that must be validated against real sensor data rather than specified in advance.

Collision risk. A hash collision, two distinct environmental states mapping to the same hash value, is a specific failure mode: INQIT would not detect a divergence between states that collide. In safety-critical domains this is not acceptable as an unanalyzed risk. Collision probability in SimHash is bounded by $(1 - \frac{\theta}{\pi})$ for angular distance θ between states, but the practical collision rate for the specific dimensional distributions encountered in a given deployment context must be measured empirically. Using multiple independent hash families per dimension (an ensemble approach) reduces collision risk multiplicatively and should be

considered for safety-critical tiers.

Dimensional sensitivity tuning. The sensitivity parameter θ_i controls how finely the hash discriminates: a high-sensitivity setting flags small changes as anomalous, appropriate for safety-critical dimensions where even small deviations matter; a low-sensitivity setting absorbs normal variation, appropriate for inherently noisy dimensions. The tuning of θ_i is not a one-time calibration — it is a deployment-context-specific design choice that the autonomous calibration process refines over time. Stage One proof-of-concept work (Section 10) must address all three of these questions empirically before INQIT is used in safety-critical deployments.

The dimensional hashes are combined into a master state hash:

$$H(t) = \Phi\big(h_1(t),\, h_2(t),\, \ldots,\, h_N(t);\, \mathbf{w}\big) \tag{8}$$

where $\mathbf{w} = (w_1, w_2, \ldots, w_N)^\top \in \Delta^{N-1}$ is the dimensional weight vector on the $(N-1)$-simplex (weights sum to one, each non-negative), and Φ is a weighted aggregation function. Both $H(t)$ and the dimensional vector $\mathbf{h}(t) = (h_1(t), \ldots, h_N(t))^\top$ are retained and inspectable.

5.2 The Hash Distance Metric

State comparison operates on hash distance. The dimensional distance for dimension i between time points t_1 and t_2 is:

$$\delta_i(t_1, t_2) = \big|h_i(t_1) - h_i(t_2)\big| \tag{9}$$

The composite state distance is the weighted Euclidean norm:

$$\Delta H(t_1, t_2) = \left(\sum_{i=1}^{N} w_i \cdot \delta_i(t_1, t_2)^2\right)^{1/2} \tag{10}$$

Dimensional alerts fire when $\delta_i(t_1, t_2) > \tau_i$ for any dimension i. System-level alerts fire when $\Delta H(t_1, t_2) > T$. Thresholds τ_i and T are calibrated by the autonomous

calibration process (Section 8).

6. Detection: The Hash Derivative System

The hash architecture described above produces a compact fingerprint of environmental state at each moment. The question is what to do with that fingerprint. The naive answer—compare each snapshot to a fixed reference and alert when they differ—fails almost immediately in a physical environment. Physical environments are always changing. A power grid's load profile at 2pm is different from its load profile at 2am. An autonomous vehicle moving through traffic is constantly changing its position, velocity, and sensor readings. Comparing to a fixed reference would trigger constant false alarms on normal operational variation.

The derivative of the hash values over time is the key insight that makes detection practical.

6.1 Why the Derivative? What It Uniquely Yields

The derivative does not ask "is this state different from the reference?" It asks "is this state *changing* in a way that indicates a problem?" This is a fundamentally different and more useful question for physical-world monitoring.

Consider what the derivative captures that static comparison does not:

Direction of change matters. A rising temperature in a cooling system and a falling temperature in the same system are both changes from a reference point, but they indicate opposite things. A derivative with sign tells you which way the system is moving, not just that it moved.

Rate of change matters. A temperature that rose one degree over an hour is within normal operational variation. The same temperature change in one second is an emergency. Static comparison cannot distinguish these, while the derivative captures the rate and therefore the urgency.

Acceleration as early warning. The second derivative, the rate of change of the rate of change, can provide an earlier warning signal by identifying acceleration toward an unsafe trajectory before an absolute threshold is crossed. A system whose temperature is rising and whose rate of temperature rise is itself increasing is approaching a critical

threshold faster than any static or first-derivative alert would capture. The second derivative catches this trajectory before the system reaches a dangerous absolute state.

Composites of derivatives localize the problem. When the master hash derivative signals an anomaly, the dimensional derivative vector tells you which specific dimension is driving the signal. This is information that no static comparison can provide: not just that something changed, but which dimension is changing, how fast, and in what direction simultaneously.

Compared to existing monitoring approaches: digital twins model the physical system rather than the AI's learned representation of it, and are typically built by the same organization as the system they monitor, which limits architectural independence. Threshold-based alerting fires when a value exceeds a fixed limit, but misses gradual drift toward that limit. Anomaly detection on raw sensor streams requires comparing high-dimensional data directly, which is computationally expensive and sensitive to normal noise.

6.2 Continuous-Time Formulation

In the theoretically complete continuous-time formulation, the instantaneous rate of change of the master hash is:

$$\dot{H}(t) \stackrel{\text{def}}{=} \frac{dH}{dt}(t) = \lim_{\Delta t \to 0} \frac{H(t) - H(t - \Delta t)}{\Delta t} \tag{11}$$

The second-order time derivative (acceleration of state change) is:

$$\ddot{H}(t) \stackrel{\text{def}}{=} \frac{d^2H}{dt^2}(t) = \lim_{\Delta t \to 0} \frac{\dot{H}(t) - \dot{H}(t - \Delta t)}{\Delta t} \tag{12}$$

For deployment contexts where the physical dynamics are governed by known differential equations (Equation 1), the relationship between $\dot{H}(t)$ and the true validity gap $\epsilon(t)$ can in principle be characterized analytically, giving theoretically grounded alert thresholds. This is an avenue for future work in bounded physical domains with known governing equations.

6.3 Discrete-Time Implementation

In practice, sensor data arrives at discrete sampling intervals Δt. The practical first-order finite difference approximation is:

$$H'(t) \approx \frac{H(t) - H(t - \Delta t)}{\Delta t} \tag{13}$$

The second-order finite difference approximation:

$$H''(t) \approx \frac{H'(t) - H'(t - \Delta t)}{\Delta t} \tag{14}$$

The dimensional derivative vector $\mathbf{h}'(t) = \big(h'_1(t), \ldots, h'_N(t)\big)^\top$ tells INQIT which dimensions are changing, how fast, and in what direction.

> **Open Question (Sampling Theory):** Discrete sampling introduces two distinct sources of error. First, finite-difference approximation error: the derivative estimate degrades as the sampling interval increases relative to the rate of change of the signal. Second, aliasing risk: the Nyquist-Shannon sampling theorem establishes that frequencies above $1/(2\Delta t)$, cannot be reconstructed without aliasing, producing false derivative signals or missing true ones. The appropriate Δt for a given deployment context is an empirical question that Stage One proof-of-concept work must address. For rapidly evolving phenomena, such as high-frequency structural vibration or fast electromagnetic transients in PNT systems, the continuous-time formulation may be necessary rather than optional.

6.4 Detection Signal Taxonomy

The derivative system produces four detection signal classes:

1. **Gradual Drift:** $H'(t)$ is persistently non-zero but below alert threshold; $H''(t) \approx 0$. Observable environmental state is changing persistently in a pattern that may indicate increasing divergence between the model's learned representation and physical reality. This is warning, not alert, but it is the most important signal to catch early. A system that is drifting gradually will eventually reach a threshold where the drift becomes operationally significant. By the time a static-comparison system would alert, the drift may have been compounding for hours.

2. **Acute Anomaly:** $H'(t)$ spikes sharply above alert threshold. An anomalous event has occurred: sensor corruption, adversarial input, physical failure, or rapid environmental change the model has not incorporated. Immediate alert.
3. **Recovery:** $H'(t)$ was elevated and is now declining toward baseline. The system is adapting to a perturbation. Monitor: do not alert unless recovery stalls. A system that recovers cleanly is behaving as expected. One that starts to recover and then stalls may indicate a deeper problem.
4. **Permanent State Change:** $H'(t)$ was elevated and has stabilized at a new non-baseline level. The observable environmental state has stabilized at a new level: either the environment has changed permanently, or the shift may indicate the model is operating on a stale picture that no longer reflects current conditions.

6.5 Dimensional Localization

Definition 6.1 (Localization Set)**.** The *localization set* at time t is:

$$\mathcal{L}(t) = \{i \in \{1, \ldots, N\} \mid |h'_i(t)| > \tau'_i\} \tag{15}$$

where τ'_i is the derivative alert threshold for dimension i.

The localization set transforms a system-level alert into an actionable diagnostic. In the July 2026 HuggingFace incident, security teams had to reconstruct more than 17,000 recorded events after the fact to understand what had happened [11]. INQIT's derivative audit trail would have surfaced the anomaly in real time and identified which dimensional hashes diverged first, which is both earlier warning and faster investigation.

6.6 Accountability: The Derivative Audit Trail

Every derivative value computed by INQIT is timestamped and retained, forming a continuous audit trail of how the system's state evolved. This constitutes the accountability record the Stanford HAI brief describes as essential: a time-stamped record of what the system perceived, the state it inferred, and the action it took [1]. INQIT's derivative record does not require access to the world model's internal weights or latent state representation, both of which remain largely uninterpretable

by current methods. It is a record of observable facts, not of model internals.

This distinction matters for legal and regulatory accountability. A record of what sensors reported, what the hash values were, and how their derivatives evolved over time is the kind of evidence that liability frameworks, regulators, and courts can actually use. A record of latent vector activations inside a neural network is not.

> **Open Question (Hash Drift and Latent-State Validity):** The foundational assumption underlying INQIT's detection architecture is that observable sensor-dimension hash drift corresponds to actual model invalidity, that is, that a diverging hash derivative signals a widening gap between the model's learned representation and the true environmental state. This mapping is the linchpin of the architecture. Without a provable or empirically validated correspondence between hash-space drift and latent-state drift, regulators and safety engineers will correctly ask: how do we know that INQIT's derivative signal actually indicates that the model is behaving unsafely, rather than merely that the physical environment changed in a way the model has correctly tracked?
>
> This question cannot be resolved analytically in general, because the model's latent state representation is not directly interpretable with current methods. The path to validation is empirical: Stage One proof-of-concept work must demonstrate, in a bounded domain with known ground truth, that hash derivative anomalies correspond to documented failure modes and that periods of low hash derivative correspond to validated correct model behavior. Until this correspondence is empirically established for a given deployment context, INQIT's derivative signals should be treated as indicators warranting human investigation, not as autonomous safety certifications. This is the most important open question in the architecture, and it is the one that determines whether INQIT can make credible safety claims rather than useful monitoring contributions.

6.7 Transferability Assessment

The simulation-to-reality gap is assessed by the *transferability index*:

$$\Psi(\text{sim}, \text{real}) = \frac{1}{T} \int_0^T \left| H'_{\text{sim}}(t) - H'_{\text{real}}(t) \right| dt \tag{16}$$

A value $\Psi \approx 0$ indicates the simulation closely tracks real-world state evolution. A large Ψ indicates systematic divergence, localized by the dimensional derivative comparison to the specific physical dimensions where the gap is largest. This is a quantified, continuous, domain-specific measure of whether the simulation is a trustworthy training ground for this deployment context, something that currently does not exist as a standardized tool.

> **Open Question (Transferability Normalization):** Equation 16 assumes simulation and deployment share comparable timescales and sensor noise characteristics. In practice, simulation environments often run faster than real time and have different noise profiles. Normalizing the transferability index across these differences is an open methodological question.

7. Successive Approximation: Building the Dimensional Framework

The most common objection to dimensional governance frameworks is that the physical world is too complex and context-dependent for any fixed dimensional framework to capture adequately in advance. This objection is correct. INQIT is designed from the ground up to address it rather than to ignore it.

7.1 The Core Challenge

Consider what it means to define the dimensions relevant to safe operation of a world model managing a mid-sized hospital's logistics. The obvious dimensions are there: equipment location, staff availability, medication inventory, bed occupancy. But safe operation in a hospital is not just a function of these quantities individually. It depends on their interactions: the ratio of available beds to incoming emergency patients, the proximity of specific equipment to the patients who need it, the alignment of staff availability with procedure schedules. These second-order relationships are not dimensions in any simple sense, and they are not obvious in advance to anyone who has not operated a hospital logistics system in real conditions.

Now multiply this across all world-model deployment contexts. Critical infrastructure, autonomous vehicles, emergency response, military logistics, each has its own set

of dimensions and interactions, many of which are only discoverable through operational experience rather than advance specification. No fixed dimensional framework, specified by anyone, will be correct on first attempt.

This is an honest description of the problem and it motivates the successive approximation approach.

7.2 Coverage as a Set-Theoretic Objective

Recall from Equation 3 that dimensional coverage is:

$$\kappa(D) = \frac{\mu(\mathcal{D} \cap \mathcal{F})}{\mu(\mathcal{F})} \tag{17}$$

The successive approximation process is an iterative algorithm for improving $\kappa(D^{(k)})$ across refinement cycles $k = 0, 1, 2, \ldots$ Each cycle produces an updated dimensional framework $D^{(k+1)}$ such that:

$$\mathbb{E}\big[\kappa(D^{(k+1)})\big] \geq \kappa(D^{(k)}) \tag{18}$$

This formalizes the successive approximation objective: each cycle is expected to improve or maintain coverage. It does not guarantee convergence to $\kappa = 1$. Complete coverage is not achievable in general, because $\mathcal{F}$ is not fully enumerable. The goal is not perfect coverage. It is improving coverage, transparently documented, with known gaps and a structured process for closing them.

7.3 Initial Dimensional Seeding

The initial dimensional framework $D^{(0)}$ is constructed through three inputs.

Domain expert specification provides the foundation. Engineers and safety specialists who have operated in the deployment domain bring prior incident data, near-miss records, regulatory requirements, and operational intuition. Their dimensional proposals are the most reliable starting point, and they carry the domain credibility that any governance framework requires to be taken seriously.

LLM-assisted dimensional enumeration accelerates coverage. Large language models, prompted with the deployment context and the expert-specified dimensions, can

generate candidate dimensions the human team has not yet identified, propose plausible value ranges, and surface edge cases from analogous domains. This is not a replacement for domain expertise. A language model that has never operated a hospital logistics system cannot know which dimensional interactions matter in practice. But it can accelerate the enumeration of candidates that human experts then evaluate. The LLM serves as a research assistant, not an authority.

Regulatory and standards mapping ensures compatibility. Existing safety standards and certification requirements for the deployment domain carry dimensional information implicitly, in the form of the quantities they require to be measured and reported. Mapping these requirements to dimensional definitions ensures that INQIT's framework does not duplicate or contradict existing compliance infrastructure, and may reveal dimensions that the expert and LLM processes would not independently identify.

7.4 Iterative Refinement Cycle

Each refinement cycle proceeds through four stages. Deployment and observation generate derivative records and alerts under real operational conditions. Gap identification examines these records for false negatives (cases where INQIT did not alert but something went wrong) and false positives (cases where INQIT alerted but nothing was wrong). False negatives indicate missing dimensions or miscalibrated thresholds. False positives indicate over-sensitive thresholds or dimensions that are proxies for something more fundamental and therefore fire on normal variation in that proxy. Dimensional refinement proposes additions, deletions, and recalibrations based on the gap analysis, with LLM assistance for candidate generation reviewed by domain experts. Calibration validation evaluates whether the updated framework produces improved signal quality across the full deployment history, not just recent cases.

7.5 Coverage Metrics

$$\mathrm{FR}(k) = \frac{\mathrm{FN}(k)}{\mathrm{FN}(k) + \mathrm{TP}(k)} \tag{19}$$

$$\mathrm{FPR}(k) = \frac{\mathrm{FP}(k)}{\mathrm{FP}(k) + \mathrm{TN}(k)} \tag{20}$$

The objective across successive cycles is to reduce $\mathrm{FR}(k)$ toward zero while maintaining acceptable $\mathrm{FPR}(k)$. The trade-off between these objectives reflects the operational reality of the deployment context. A nuclear facility accepts very high false positive rates in order to drive false negatives toward zero. A commercial logistics operation may accept higher false negative rates in order to avoid alert fatigue that causes operators to ignore the system. The calibration process learns this trade-off from operational experience rather than having it pre-specified.

8. Autonomous Calibration: Keeping the Safety System Honest

A detection system calibrated once and then deployed statically accumulates blind spots as the deployment environment evolves around it. A hospital logistics system tuned in January may be badly miscalibrated by December, as patient populations change, new equipment is installed, and workflows shift. A power grid monitor calibrated for summer demand profiles will behave unpredictably during a polar vortex that drives demand patterns outside the calibration range.

INQIT addresses this through an autonomous calibration process: a dedicated AI-driven function whose purpose is not to monitor the world model but to monitor INQIT itself. This is a critical architectural distinction. A safety system that cannot evaluate its own performance is a safety system that degrades silently. The autonomous calibration process makes INQIT's performance visible, traceable, and improvable over time.

8.1 The Calibration Loss Function

Let $C \in \mathbb{R}^M$ denote the calibration parameter vector (threshold values τ_i, T, and τ_i' for all dimensions). The calibration loss function captures the operational cost of both error types:

$$\mathcal{L}(C, \mathcal{H}) = \lambda_{\mathrm{FN}} \cdot \mathrm{FR}(C, \mathcal{H}) + \lambda_{\mathrm{FP}} \cdot \mathrm{FPR}(C, \mathcal{H}) \tag{21}$$

where $\mathcal{H}$ is the deployment history, and $\lambda_{\mathrm{FN}}, \lambda_{\mathrm{FP}} \geq 0$ are cost weights set by domain

experts reflecting the relative cost of each error type in the deployment context.

The cost weights λ_{FN} and λ_{FP} encode a policy judgment, not a mathematical choice. In a nuclear facility, λ_{FN} is very large relative to λ_{FP}: missing a real anomaly is catastrophically worse than a false alarm that triggers an unnecessary inspection. In a commercial warehouse routing system, the trade-off may be closer to balanced: false alarms have real operational cost, and some false negatives can be tolerated if the system is generally reliable. The calibration process optimizes for the right trade-off in the right context. It does not impose a universal standard that is appropriate for no specific context.

8.2 The Calibration Update Rule

The calibration loop runs continuously, proposing threshold adjustments via gradient descent on the calibration loss:

$$C_{k+1} = C_k - \alpha \cdot \nabla_C \mathcal{L}(C_k, \mathcal{H}) \tag{22}$$

where $\alpha > 0$ is the learning rate.

> **Open Question (Differentiability and Incomplete Outcome Records):** The gradient $\nabla_C \mathcal{L}$ requires $\mathcal{L}$ to be differentiable with respect to C. In practice, FR and FPR are step functions of the threshold parameters, making direct gradient computation ill-defined. Surrogate smooth approximations (sigmoid relaxations of the threshold indicator functions) are standard in related settings but introduce bias. Additionally, the outcome record $\mathcal{H}$ is often incomplete in real deployments: not every near-miss is recorded, and some events are ambiguously labeled. A logistics incident that was avoided by a human override may or may not have been a genuine failure mode. The record is ambiguous. How much incompleteness the calibration loop can tolerate before its proposals become unreliable is an open empirical question. Candidate approaches include Bayesian updating with uncertain labels and semi-supervised methods that infer outcome labels from derivative patterns.

A Hypothetical Example

To make the calibration loop concrete, consider a warehouse robotics deployment that has been running for ninety days. The outcome record contains eighteen documented events: fourteen false positives (INQIT alerted but no failure occurred) and four false negatives (a near-miss occurred that INQIT did not flag). Fourteen of the false positives were triggered by dimension d_3, the obstacle proximity hash, during a period when the facility was reorganizing its layout, normal operational variation that exceeded the threshold. The calibration loss function, with cost weights reflecting that false negatives are three times more costly than false positives in this context, computes a gradient that pushes τ_3 upward (reducing sensitivity to reduce false positives) while pushing τ_2, the velocity hash threshold, downward (increasing sensitivity to catch the near-misses, which involved unexpected velocity changes). The calibration loop proposes: raise τ_3 by 12%, lower τ_2 by 8%. A human reviewer examines the four near-miss records, confirms that velocity anomalies preceded each one, and approves the proposal. The updated thresholds take effect for the next operational period. Documented false negatives fall from four in the initial period to one in the following thirty days. This is the calibration loop working as designed: learning the right trade-off for this specific deployment context from operational experience rather than having it pre-specified.

8.3 Human Oversight of the Calibration Process

The autonomous calibration process proposes changes but it does not implement them unilaterally. All proposed threshold adjustments and dimensional changes above a defined significance threshold require human review and approval before taking effect.

This is not a bureaucratic constraint. It is a substantive governance requirement. The calibration process is optimizing against a loss function, and any optimization process can find unexpected minima. A calibration proposal that dramatically reduces false positives by raising all thresholds to the point where only catastrophic events would trigger an alert is technically a reduction in FPR, but it is not a reduction that any reasonable operator would approve. Human review of proposals is the check against this kind of pathological optimization.

The calibration process's proposals, confidence scores, and supporting evidence are

logged in full and available for regulatory inspection. A regulator asking "why did INQIT set this threshold at this value?" receives a complete, traceable answer: the deployment history that supported the proposal, the cost weights that governed the optimization, and the human approval that authorized the change.

Open Question (Calibration Failure Modes): Any self-adjusting safety system requires explicit analysis of the ways the adjustment process itself can fail. Three failure modes warrant particular attention:

Outcome record bias. The calibration loss function optimizes against a labeled outcome record. If that record is systematically biased, the calibration loop will converge on a parameter vector that minimizes labeled losses while missing unlabeled ones. In safety-critical deployments, near-misses that were resolved by human intervention may not be recorded as failures, causing the calibration process to underweight the thresholds that would have caught them.

Distribution shift. The calibration loop learns from historical deployment data. If the deployment environment undergoes a structural shift, the historical data may no longer be representative, and a calibration parameter vector that was well-tuned to past conditions may be systematically miscalibrated for current ones. The gradual drift detection signal (Section 6.4) is designed to surface this, but does not by itself prevent a calibration loop from following a drifting baseline toward a new, incorrect equilibrium.

Convergence to local minima. The gradient descent calibration update rule (Equation 22) is not guaranteed to converge to a globally optimal parameter vector. For non-convex calibration loss landscapes, the loop may stabilize at a local minimum that appears acceptable by the loss metric but is inadequate for safety. Running multiple calibration instances from different initializations and comparing their converged parameters is a candidate mitigation, but this has not been validated empirically.

These failure modes reinforce the requirement for human review of all calibration proposals above a defined significance threshold. The review process is not a formality; it is the mechanism by which calibration failure modes are caught before they propagate into operational thresholds.

9. Deployment Contexts and Priority Domains

The INQIT architecture is general, but its most urgent applications are in deployment contexts where the cost of world-model failure is highest, the simulation-to-reality gap is most consequential, and the adequacy of current governance is most clearly insufficient. This section describes five priority domains in enough specificity to illustrate what INQIT's dimensional framework and derivative detection would actually look like in practice.

9.1 Critical Infrastructure

Power grids, water systems, transportation networks, and hospital logistics systems are managed by AI systems that make consequential decisions in real time. A power grid world model making load-balancing decisions does not produce a recommendation for human review. It directs switching events that happen in milliseconds. A hospital logistics AI does not suggest that medication X should be prepared; it routes it. In both cases, the question is not whether the output seems reasonable. It is whether the model's picture of the system state is accurate.

For a power grid deployment, INQIT's dimensional framework would include load by zone and time of day, frequency deviation (the real-time signal of supply-demand balance), voltage stability margins across key nodes, inter-area power flows, and temperature-dependent demand forecasts. The derivative system would distinguish between normal diurnal load variation (gradual, predictable, low derivative magnitude) and demand spikes that exceed the model's predictive capacity (sharp derivative increase, localized to specific load zones). The permanent-state-change signal would flag cases where a new baseline emerged that the model had not incorporated, for example, a large industrial load coming online that permanently shifts zone demand profiles.

The failure mode INQIT is specifically designed to catch: a model that learned its demand patterns during normal operating conditions and is now making dispatch decisions during an extreme weather event that drives demand outside its training distribution. The model does not know it is wrong. It is confident and wrong. INQIT's derivative would show the hash values for demand-forecast dimensions diverging from the hash values for actual-load dimensions, signaling that observable demand conditions

are diverging from the model's forecast dimensions in ways that warrant investigation before that divergence produces a dispatch decision that cascades.

9.2 Autonomous Vehicles and Robotics

For an autonomous vehicle, the world model is the system's continuous picture of its immediate environment: where other vehicles are, how fast they are moving, what the road geometry is, what the surface conditions are, where pedestrians are and which direction they are moving. Failure of this model is not a software error in the conventional sense. It is the vehicle acting on a picture of the world that is wrong.

INQIT's dimensional framework for an autonomous vehicle deployment would include positional certainty (the spread of the vehicle's position estimate), obstacle proximity and velocity in multiple zones, road surface condition estimates, semantic scene complexity (a measure of how many interacting elements the model is tracking), and sensor health metrics for each input stream. The derivative system would flag cases where the model's positional certainty is decreasing rapidly (the vehicle is losing track of where it is), where obstacle velocity estimates are changing faster than physics would permit (the model is confused about a sensor artifact), or where semantic complexity has spiked beyond the model's demonstrated reliable operating range.

The transferability index Ψ is particularly important for autonomous vehicles. The simulation environments in which these systems are trained are, by design, well-controlled and representative. The real world is neither. Measuring Ψ continuously in deployment provides a quantified, running answer to the question that regulators, insurers, and the public most need answered: is this system operating in conditions sufficiently similar to those in which it was validated?

9.3 Positioning, Navigation, and Timing (PNT) Systems

PNT infrastructure is a priority domain for INQIT because it is both critical to an enormous range of downstream systems and specifically vulnerable to the kind of adversarial sensor corruption INQIT's stochastic game formulation (Equation 4) is designed to address.

GPS spoofing, the injection of false GPS signals to corrupt a receiver's position estimate, is a documented, demonstrated threat across military, aviation, maritime,

and civilian contexts. A world model that relies on GPS signals without cross-validation is vulnerable to a spoofing attack that makes it believe it is somewhere it is not. The attack is invisible to the model: the signal looks valid, the position estimate is confident, and the model acts on a picture of its location that is systematically wrong.

INQIT's dimensional framework for PNT would include position from each independent receiver, velocity consistency across receivers, timing accuracy and drift, carrier-to-noise ratio per satellite, geometric dilution of precision (a measure of satellite geometry quality), and cross-receiver consistency hashes that compare position estimates across redundant, independently sourced receivers. The derivative system would flag rapid changes in carrier-to-noise ratio that are not explained by satellite geometry changes (a signature of spoofing or jamming), divergence between independent receiver position estimates (a direct indicator of one receiver being manipulated), or timing drift that is inconsistent with satellite clock models.

Cross-receiver consistency checking is the natural INQIT implementation for PNT: comparing the hash values for position and timing across multiple independent receivers, with the localization set identifying which receiver's hash is diverging from consensus. An adversary who can spoof all receivers simultaneously faces a much harder problem than one who only needs to fool a single receiver.

9.4 Emergency Response and Crisis Management

Emergency response is a domain where world-model errors have direct, immediate human consequences and where the data environment is inherently degraded. A crisis produces exactly the conditions in which trained models are most likely to fail: novel situations, missing or unreliable data, rapidly evolving conditions, and time pressure that prevents deliberate human override.

A world model supporting wildfire response, for example, maintains a picture of fire perimeter, wind patterns, road accessibility, evacuation status by zone, and resource availability across air and ground assets. Each of these dimensions degrades in a crisis: roads that were passable become impassable, wind models become unreliable as local conditions dominate, resource counts change faster than data infrastructure can track. A model that was accurate before the crisis is less accurate during it, and the rate of degradation is itself a critical signal.

INQIT's derivative system is well-suited to this environment. The gradual drift signal would catch early signs that environmental conditions are diverging from training scenarios in ways that may indicate model accuracy degradation. The acute anomaly signal would catch sudden environmental changes that the model has not yet incorporated. The localization set would identify which dimensions have become unreliable, giving crisis managers specific, actionable information about which parts of the model's picture to trust and which to verify independently.

9.5 Defense and National Security

The Stanford HAI brief identifies world models as carrying major national security implications across autonomous operations, logistics under contested conditions, and intelligence analysis. The brief identifies three specific risks: systems deceived by corrupting their picture of the environment, false readiness from synthetic training environments, and miscalculation from machine-speed operations with limited human oversight [1].

INQIT's architecture addresses each directly. Adversarial sensor corruption is the focus of the stochastic game formulation and the cross-sensor consistency framework. False readiness from synthetic training is what the transferability index is designed to measure: the degree to which a system's behavior in deployment matches its behavior in the environments in which it was validated. And the derivative audit trail addresses miscalculation accountability: when a system operating at machine speed makes a decision that has consequences, the derivative record provides a timestamped account of the model's environmental picture at the moment of decision, the information base on which the action was predicated.

One national security consideration worth naming explicitly: INQIT's architecture is designed to be architecturally independent of the systems it monitors. This means it can be operated by a party other than the system developer, which is a prerequisite for any meaningful external oversight of AI systems used in classified or sensitive national security contexts.

10. Development Roadmap and Open Questions

INQIT is a conceptual architecture. The following roadmap identifies development stages and consolidates the open research questions raised throughout the paper.

10.1 Stage One: Proof of Concept in a Bounded Domain

The first implementation should target a deployment context where relevant physical dimensions are well-understood, ground truth is available for calibration, and the consequences of false positives are manageable. Industrial robotics in a controlled warehouse environment is a strong candidate: relevant dimensions (position, velocity, obstacle proximity, load, temperature) are measurable and well-defined. The outcome record is available and a false positive costs a brief operational pause rather than a life. Stage One must answer: Does the LSH architecture produce stable, calibratable hash values for real sensor data? Does the derivative system produce early warning in advance of known failure modes? What is the practical Δt for this environment? How many refinement cycles are required before coverage is adequate for reliable operation?

10.2 Stage Two: Calibration Process Validation

The autonomous calibration process must be validated in shadow mode, computing proposals without implementing them, and evaluating proposal quality against human expert judgment. The objective is to establish convergence of the calibration loop and to quantify how much deployment history is required for proposals to reach acceptable confidence levels.

10.3 Stage Three: Multi-Domain Generalization

Stage Three tests whether the INQIT architecture generalizes across deployment contexts without requiring complete reconstruction of the dimensional framework. A framework developed for warehouse robotics should be adaptable to emergency response or infrastructure management with acceptable effort. The degree of generalization determines INQIT's scalability as a governance platform.

10.4 Consolidated Open Research Questions

Open Question 1 (Approximation Error): What is the relationship between INQIT's finite-difference derivative approximation and the true validity gap $\epsilon(t)$ of Equation 2? Under what conditions on the sampling interval Δt and the dynamics of $\mathcal{E}(t)$ is the approximation adequate for safety-critical monitoring?

Open Question 2 (Dimensional Completeness Bounds): What theoretical bounds exist on the coverage $\kappa(D)$ of Equation 3 given a partial characterization of the failure-mode set $\mathcal{F}$? Under what conditions does the successive approximation process converge, and to what value of κ?

Open Question 3 (Adversarial Robustness): How does INQIT perform against the adversarial objective of Equation 4? What are the conditions under which cross-sensor consistency checking adequately mitigates adversarial sensor corruption? What is the game-theoretic equilibrium of the adversary-INQIT interaction?

Open Question 4 (Calibration Convergence): Does the calibration update rule of Equation 22 converge to a stable parameter vector C^*? What are the conditions on the deployment history $\mathcal{H}$ and the cost weights $\lambda_{\text{FN}}, \lambda_{\text{FP}}$ required for convergence?

Open Question 5 (Multi-Agent Attribution): Where multiple world models operate in the same physical environment, how are anomalies attributed to specific models? The localization set $\mathcal{L}(t)$ identifies which dimensions are anomalous. It does not identify which model caused the anomaly when multiple models share the same dimensional environment.

11. Conclusion: The Measurement Science Gap

The governance gap the Stanford HAI brief identifies is not primarily a policy gap. Policies exist. Standards are being written. Commissions are being proposed. What is missing is the measurement science that makes those policies, standards, and commissions meaningful, the measurement science that allows a policymaker to ask

"is this world model safe for this deployment?" and receive an answer grounded in continuous, real-world evidence rather than pre-deployment certification that may have expired the moment the model encountered an environment its training did not represent.

Pre-deployment benchmarks provide a snapshot and the world moves on. Digital twins provide depth for known systems but they cannot validate arbitrary learned environments. Runtime verification provides strong guarantees for formally specifiable properties, but world-model safety is not yet fully formalizable. Standards bodies provide shared frameworks; AI capability compounds faster than certification cycles.

INQIT is a proposed conceptual architecture for closing this measurement science gap. It does not make world models safe. It proposes a measurement architecture for making emerging evidence of environmental invalidity detectable, locatable, and reconstructable — in real time, at machine speed, across the physical dimensions that matter for the deployment context in question — so that model validity can be assessed continuously rather than inferred from a pre-deployment snapshot.

The architecture is offered as a public framework, not proprietary, not patented, offered for research community engagement, critique, and implementation. INQIT is a starting point for the measurement science the Stanford HAI brief calls for: a structured hypothesis about what physical-world AI governance measurement science needs to look like, with five explicit open questions identifying where the empirical work remains undone.

It is worth noting that INQIT is designed to be policy-compatible rather than policy-dependent. Its derivative audit trail provides the accountability record that liability frameworks, regulatory agencies, and congressional oversight require, without waiting for those frameworks to be finalized. Its transferability index provides the deployment-readiness evidence that a standards body or certification regime would need to make meaningful deployment decisions, rather than relying on vendor claims. And its successive approximation process is structurally compatible with the iterative, scenario-specific regulatory approach that the European Union has demonstrated works better than fixed upfront standards: governance and measurement science improve together, each informing the other, without either needing to wait for the other to be complete. INQIT does not require a policy mandate to be useful. But it is built to support one when it arrives.

Standards bodies and certification regimes are typically the solve-later approach. INQIT can be a framework for solving now.

References

[1] Zhang, D., Wald, R., Adeli, E., Cryst, E., Ho, D.E., Meinhardt, C., Wu, J., Zegart, A., and Li, F.F. (July 2026). *The World Model and Spatial Intelligence Era: Governing AI Beyond Language.* Stanford HAI Issue Brief.

[2] Bengio, Y. et al. (February 2026). *International AI Safety Report 2026.* Université de Montréal / Mila.

[3] Hassabis, D. (July 2026). "A Framework for Frontier AI and the Dawning of a New Age." Google DeepMind.

[4] Future of Life Institute. (July 2026). *AI Safety Index, Summer 2026.* futureoflife.org

[5] Frazier, K. and Reddie, A. (July 2026). "The Government Is Choosing AI Models. Who Chooses Their Values?" *AI Frontiers.*

[6] Barton-Cooper, R. and Gleave, A. (August 2026). "AI Jailbreak Disclosure Is Broken. Here's How to Fix It." *AI Frontiers.*

[7] Weil, G. (July 2026). "Don't Let AI Developers Hire Their Own Referees." *AI Frontiers.*

[8] Li, F.F. (June 2026). "A Functional Taxonomy of World Models." World Labs.

[9] METR (Model Evaluation and Threat Research). (2026). Independent evaluations of frontier models. metr.org

[10] Holt, T.D. (2026). Internal validation study: deterministic scoring architecture applied to language AI exchanges. Patent pending, SurfWax Inc.

[11] de Gregorio, I. (August 2026). "AI's First Autonomous Cyber Attack." Medium.

[12] Nov Tech. (July 2026). "He Asked for an Emergency Stop Button. He Was the First One It Hit." Medium.

[13] The Economist. (September 2025). "Why AI systems may never be secure, and what to do about it."

[14] Charikar, M. (2002). "Similarity estimation techniques from rounding algorithms." *Proceedings of STOC '02*, pp. 380–388. ACM.

[15] Indyk, P. and Motwani, R. (1998). "Approximate nearest neighbors." *Proceedings of STOC '98*, pp. 604–613. ACM.

[16] Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V.S. (2004). "Locality-sensitive hashing scheme based on p-stable distributions." *Proceedings of SCG '04*, pp. 253–262. ACM.

[17] Shannon, C.E. (1949). "Communication in the presence of noise." *Proceedings of the IRE*, 37(1), 10–21.

[18] Temam, R. (2001). *Navier-Stokes Equations: Theory and Numerical Analysis.* AMS Chelsea Publishing.

A. Mathematical Notation Summary

Symbol	Definition
$\mathcal{E}(t)$	True environmental state at time t
$\hat{\mathcal{E}}(t)$	World model's estimated state at time t
$\epsilon(t)$	Validity gap: $\left\|\mathcal{E}(t) - \hat{\mathcal{E}}(t)\right\|$
N	Number of physical dimensions in the framework
$D = \{d_1, \ldots, d_N\}$	Set of physical dimensions
$m_i(t)$	Measurement function for dimension i at time t
$\text{LSH}(\cdot, \theta_i)$	Locality-sensitive hash with sensitivity θ_i
$h_i(t)$	Dimensional hash for dimension i at time t
$\mathbf{h}(t)$	Dimensional hash vector $(h_1(t), \ldots, h_N(t))^\top$
$H(t)$	Master state hash at time t
$\Phi(\cdot; \mathbf{w})$	Hash fusion function with weight vector $\mathbf{w}$
$\delta_i(t_1, t_2)$	Hash distance for dimension i
$\Delta H(t_1, t_2)$	Weighted Euclidean composite state distance
τ_i	Static alert threshold for dimension i
T	Composite system-level alert threshold
$\dot{H}(t)$	Continuous-time first derivative of master hash
$\ddot{H}(t)$	Continuous-time second derivative of master hash
$H'(t)$	Finite-difference first-order derivative
$H''(t)$	Finite-difference second-order derivative
Δt	Discrete sampling interval
$\mathcal{L}(t)$	Localization set: dimensions exceeding thresholds
$\mathcal{F}$	Set of safety-critical failure modes
$\mathcal{D}$	Set of failure modes detectable by framework D
$\kappa(D)$	Dimensional coverage: $\mu(\mathcal{D} \cap \mathcal{F})/\mu(\mathcal{F})$
$\Psi(\texttt{sim}, \texttt{real})$	Transferability index
C_k	Calibration parameter vector at cycle k
α	Calibration learning rate
$\mathcal{L}(C, \mathcal{H})$	Calibration loss function
$\lambda_{\texttt{FN}}, \lambda_{\texttt{FP}}$	Cost weights for false negative / false positive errors
$\texttt{FR}(k)$	False negative rate at refinement cycle k
$\texttt{FPR}(k)$	False positive rate at refinement cycle k
$\mathcal{A}$	Adversary's action space (sensor corruption)