

Contents

0. Front Matter.....	3
0.1 Title, Author, and Project Identifiers.....	3
0.2 Version, Date, and Scope of This Document	4
0.3 On-Chain and Cryptographic Record.....	5
0.4 External Attachments and How to Use Them	5
1. Framework Overview.....	7
1.1 Purpose of the Framework.....	7
1.2 Core Claims and Non-Goals.....	8
1.3 Layered Structure of the Theory	10
2. Ontology and Data Structures.....	11
2.1 Present-Act Ontology.....	11
2.2 Present-Moment Sphere (PMS)	13
2.3 Inner and Outer Networks (IN and ON)	15
2.4 Collective Spheres (CS) and Frames.....	16
2.5 Context Levels (-2, -1, 0, +1, +2, +3)	18
2.6 Qualia-First Encoding and the Feature Alphabet Ξ	20
3. Formal Core (V1) – Minimal Structural Spec	23
3.1 Primitive Objects and Carriers	23
3.2 Operator Alphabet and Tick Algebra	24
3.3 Ledger and Arrow of Time	27
3.4 Invariant Interval from Flip Counts	30
3.5 Fractal Inner Geometry and Pivot Function ($g(D)$)	32
3.6 Present Plane and Structural Born Rule	35
3.7 Context Ladder ($D(n)$) and Memory Dimension.....	38
3.8 Terminology Note: “Division-by-Zero Operator”	41
4. Present-Act Engine (V2) – Algorithmic Specification.....	42
4.1 Engine State at a Site (k)	42
4.2 Feature Alphabet Ξ and Hinge Equality	45

4.3 Engine Cycle: High-Level Pipeline	47
4.4 Feasibility Gates (Non-Gravity)	53
4.5 PF/Born Ties-Only Rule – Algorithmic Detail	57
4.6 Implementation Principles (Finiteness, Locality, Curve-Ban, Auditability)	61
5. Context Level (CL) Ladder and Hinge Scales	64
5.1 Definition of the 6-Band Ladder	64
5.2 Derivation of the Universal Geometric Mean (UGM).....	67
5.3 Temporal Hinge (T^*) and Relation to Budgets and Context	71
5.4 CL Evidence Summary (Non-Narrative)	74
6. Gravity as Feasibility Geometry and Gravitational Amplitude χ	77
6.1 Gravity in V1: Hinge and Pivot Profile	77
6.2 V2.1 Gravity Implementation: ParentGate and Shells.....	79
6.3 Definition and Derivation of Gravitational Amplitude χ	83
6.4 Comparison to GR Metric Amplitude and Factor ≈ 1.8 Discrepancy	87
6.5 Gravitational Budgets and Engine Constraints	90
7. Quantum Structure and Measurement	93
7.1 Structural Quantum Mechanics in V1	93
7.2 Engine-Side Quantum Behaviour	95
7.3 Measurement and Framing in Collective Spheres	98
8. Simulations and Evidence Programme	101
8.1 Simulation Philosophy and Reproducibility	101
8.2 V1 Simulation Families (Structural Tests)	103
8.3 V2 Engine Simulation Families	105
8.4 Limitations and Open Simulation Directions	109
9. Implementation and Reproducibility.....	112
9.1 Code Archive Overview	112
9.2 Engine Reconstruction Procedure (Step-By-Step)	114
9.3 Manifests and Parameter Tables	118
9.4 Diagnostics and Verification Tools.....	121

10. Terminology, Notation, and Accessibility.....	126
10.1 Glossary of Core Terms	126
10.2 Operator Names and Historical Labels	130
10.3 Symbol Index.....	131
11. Explicit Claim Set and Failure Modes	136
11.1 Structural Claims.....	136
11.2 Engine Claims	138
11.3 Empirical and Phenomenological Claims	141
11.4 Failure Modes and Potential Refuters	144

0. Front Matter

0.1 Title, Author, and Project Identifiers

This defensive publication records the condensed core of the Absolute Relativity / Overall V2 framework and its associated artefacts.

Document title:

Absolute Relativity / Overall V2 Theory – Defensive Publication (Condensed Core Specification)

Author / inventor:

Kent Nimmo

Project name:

Absolute Relativity / Overall V2 Theory

Contact email:

absoluterelativityproject@gmail.com

Project website:

absoluterelativity.org

In addition to the textual and archival identifiers, the project is associated with specific on-chain contract addresses, which serve as canonical identifiers and participation markers for the framework.

Ethereum (ERC-20) identity:

Token Contract: 0xAacCd7bA616405C184335F193fEf080fC982921F

Project Wallet (Deployer): 0x1F06ea3554aE665e713a637eD136a5065C9cD787

Verification Transaction:

0x9fce2bfdbb96fad2c66832e5771a0f40b1ff43213227d35870bf1616274b3c

Solana (SPL) identity:

Token Mint Address: ARafKuCqRgszXZWjYGWyBT7GnLZkyiaXQd1YjXC1x224

Project Wallet: 7mik22AsVKX2ueqSWHCD8HBMpcfEMhbKU85xYoaxCKN

These identifiers match the v0.9 source bundle already recorded via the cryptographic archive and on-chain hash, and should be used whenever referring to this defensive publication and its attached materials as a unified specification of the Absolute Relativity / Overall V2 framework.

0.2 Version, Date, and Scope of This Document

Version label:

Defensive Publication v1.0 (Condensed Core Specification based on Overall V2 – v0.9)

Underlying source bundle:

Overall V2 – v0.9 (archived as “A.R. V2 – v0.9.zip” and recorded on-chain via its cryptographic hash).

This document serves as the **condensed core specification** of the Absolute Relativity / Overall V2 framework for defensive-publication purposes. It is derived directly from the v0.9 source bundle (including the nine main documents and associated code/archive) and is intended to:

- State the core ontology, formal structures, engine design, context-level ladder, and key constructions (including hinge scales and gravitational amplitude χ) in a compact, enabling form.
- Describe the essential algorithms and constraints tightly enough that a person skilled in the art could reconstruct a functionally equivalent implementation of the present-act engine and its main tests.
- Explicitly reference the attached v0.9 documents and code archive as the **canonical, more detailed layer** wherever additional context, proofs, or extended evidence are needed.

This is **not** a philosophical essay, promotional document, or investment communication.

Narrative, ethical, and motivational material present in the longer writings has been minimized or removed here. Where such content remains, it is included only insofar as it is necessary to understand the structure and intended use of the framework.

In any case of tension or ambiguity between this condensed specification and the attached v0.9 materials, the attached materials (the nine documents and the code/archive in the v0.9 bundle) are to be treated as the authoritative source for the state of the theory and implementation details at the v0.9 snapshot.

0.3 On-Chain and Cryptographic Record

The canonical source bundle for this framework is archived as:

- **Artifact name:** A.R. V2 - v0.9.zip

This archive contains the v0.9 snapshot of the Absolute Relativity / Overall V2 framework, including:

- The nine main theory and evidence documents (big picture, philosophical underpinnings, V1 formal framework, V1–V2 bridge, V2 engine specification, context-level framework, V1 simulations, V2 simulations, core evidence narrative).
- The present-act engine source code and associated manifests.
- Simulation scripts and related supporting materials referenced in those documents.

The integrity of this archive is fixed by the following cryptographic hash:

- **Cryptographic hash (SHA-256):**
2c97e4e15227ad51b7ba73bef412c1cb16e820479b7acd3e4b56d78c19edfdd9

This hash has been written to the Ethereum blockchain from the project wallet associated with the Absolute Relativity / Overall V2 project. The on-chain record serves only as a timestamped integrity marker for the archive; it does not itself contain the archive contents.

To verify that any given copy of A.R. V2 - v0.9.zip is the canonical v0.9 bundle referred to in this defensive publication, a verifier can:

1. Obtain the archive file A.R. V2 - v0.9.zip.
2. Compute its SHA-256 hash using any standard tool.
3. Confirm that the resulting hash string matches the value shown above.

If the computed hash matches, the archive is byte-for-byte identical to the v0.9 source bundle that underlies this defensive publication, and all references in this document to “the v0.9 bundle” or “the attached materials” apply to that archive.

0.4 External Attachments and How to Use Them

This defensive publication is designed to be read together with a set of attached documents and archives that make up the v0.9 source bundle of the Absolute Relativity / Overall V2 framework. This document provides the condensed core; the attachments provide full derivations, extended argumentation, simulation details, and code.

The principal attached documents are:

- (0) Overall V2 – v0.9
Front-matter and file map for the v0.9 bundle. Describes how the nine main documents and code/archive fit together and how they should be read.
- (1) Big Picture
High-level motivation and “big picture” orientation: why the framework is constructed, what problems it is addressing, and how the main components fit conceptually.
- (2) Philosophical Underpinnings V2
Concise philosophical core: present-act / qualia-first ontology, the notion of one infinite Present with many versions, and how time, context levels, and fractal nesting are interpreted at the conceptual level.
- (3) V1 – Formal Framework
Formal mathematical backbone: PMS/IN/ON/CS objects, tick and operator algebra, ledger and invariant interval, fractal inner geometry, pivot function, context ladder profile, and structural Born-rule construction.
- (4) Bridge (V1–V2)
Mapping between V1 (formal) and V2 (engine) views. Shows how the abstract V1 objects and operators correspond to the concrete V2 data structures, gates, and update rules.
- (5) V2 – Present-Act Engine
Detailed specification of the discrete present-act engine: state representation, feature alphabet, hinge equality, feasibility gates, ratio-lexicographic acceptance, PF/Born ties-only rule, budgets, and implementation constraints.
- (6) Context Level Framework
Context-level ladder in explicit physical scales: definition and justification of the -2 , -1 , 0 , $+1$, $+2$, $+3$ bands, hinge scales (UGM and temporal hinge), and the role of geometric-mean pivots.
- (7) V1 Simulations
Simulation record for V1-era tests: operator algebra checks, fractal/pivot calibration, gauge-like modules, measurement/classicalization attempts, and their outcomes (pass/obstructed/pending).
- (8) V2 Simulations
Simulation record for V2-era tests: engine sanity checks (locality, SR-like budgets, no-signalling), matter-addition suites, gravity/feasibility scenes, and T-series applications to external data.
- (9) Core Evidence Narrative
Condensed evidence spine: cross-scale hinge structure (UGM, CNS size band, perceptual resolution), construction and use of χ , and summary of observational/simulation matches and tensions.

In addition to these nine documents, the v0.9 archive includes:

- Present-act engine source code and supporting modules.
- Example manifests specifying band boundaries, feature resolutions, and gate parameters.
- Simulation scripts, configuration files, and selected result datasets and plots.

All of the documents and materials listed above are **incorporated here by reference** as part of this defensive publication. In any case where a concept, construction, or algorithm is summarized here, the corresponding attached material should be treated as the authoritative, more detailed exposition of the same item at the v0.9 snapshot.

Recommended use:

- Use this condensed specification as the primary reference for the core ontology, structures, and algorithms.
- When additional detail, proofs, derivations, or empirical context are needed, consult the corresponding attached document(s) and, where relevant, the code and manifests in the v0.9 archive.

1. Framework Overview

1.1 Purpose of the Framework

The purpose of this framework is to specify a coherent, qualia-first present-act theory in which the primitive units are discrete present-acts, and in which the structures that appear as spacetime, matter, quantum behaviour, and gravity in conventional physics arise as derived, emergent patterns of those present-acts. This defensive publication records the systems and methods by which that framework is defined and implemented, in a form intended to be enabling for reconstruction by a person skilled in the art.

Concretely, the framework aims to:

- Replace “matter in spacetime” as the primitive ontology with discrete present-acts having inner (qualia) and outer (world) structure.
- Provide a formal core (V1) that encodes the constraints on how present-acts can be organized: the present-moment sphere (PMS), IN/ON split, operator algebra, invariant-interval structure, and context ladder.
- Provide an explicit discrete engine (V2) that realizes those constraints as a stepwise update rule over finite data structures, with a finite feature alphabet, feasibility gates, and a PF/Born ties-only rule for stochastic resolution.
- Embed this engine within a context-level ladder (CL) that ties discrete present-acts to specific physical scale bands (-2 , -1 , 0 , $+1$, $+2$, $+3$) and hinge scales such as the spatial Universal Geometric Mean (UGM) and a temporal hinge.
- Implement gravity as a feasibility geometry—via a family of radial gates and a gravitational amplitude χ —so that familiar weak-field gravitational effects appear as statistical properties of surviving histories, rather than as fundamental curvature of a background manifold.
- Record a simulation and evidence programme that tests whether this structure can reproduce core qualitative and quantitative features associated with special relativity,

quantum mechanics, gravitational lensing, rotation curves, and cross-scale biological/physical clustering.

The goal is not to provide a complete, final physical theory, but to disclose a structurally well-defined, qualia-first framework and its concrete engine, along with specific constructions (including UGM and χ) and tests that follow from it.

1.2 Core Claims and Non-Goals

This defensive publication puts on record a specific set of structural and algorithmic claims about the Absolute Relativity / Overall V2 framework. These claims are made at the level of systems and methods, not at the level of a finished empirical theory of everything.

Core structural and algorithmic claims

This document asserts the existence and specification of:

- **A qualia-first present-act ontology**
 - The primitive units are discrete present-acts, each with an inner (qualia) side and an outer (world) side.
 - “Matter in spacetime” is treated as a derived pattern of these present-acts, not as the primitive substrate.
- **A formal core (V1)**
 - A present-moment sphere (PMS) structure with IN/ON split and optional collective spheres (CS).
 - A tick and operator algebra (including Renew, Sink, Trade, Sync, and a hinge projection operator) acting on PMS/CS carriers.
 - A discrete invariant-interval structure built from flip-counts, inducing a cone-like causal structure.
 - A context ladder with a dimension profile ($D(n)$) and pivot function ($g(D)$), used to define hinge behaviour and band structure.
 - A present-plane construction with a structural Born-style rule linking amplitudes to inner basin measures.
- **A discrete present-act engine (V2)**
 - Engine state at each tick (k) represented by finite world and qualia records ((W_k, Q_k)) and typed budgets ($(\Delta \tau, \Delta t, \Delta x)$).
 - A finite feature alphabet (\mathcal{X}_i) and fixed feature maps from (W_{k+1}) and (Q_k) into (\mathcal{X}_i) , with hinge equality as the only admissibility condition at the PMS boundary.
 - A sequence of feasibility gates (including time, structural, context, and gravity/ParentGate gates), a ratio-lexicographic acceptance rule, and a PF/Born ties-only procedure that introduces randomness only when structural ties remain.
 - Implementation constraints including finiteness of candidate sets, strict locality/no-skip updates, a curve-ban on continuous control parameters, and a manifest-plus-audit structure that fixes and checks engine configuration.
- **A context-level ladder and hinge scales**

- A six-band context ladder (-2, -1, 0, +1, +2, +3) assigned to specific physical scale ranges (nano/biomolecular, micron/cellular, UGM band, Earth-surface, galactic disk, cosmic shell).
- A Universal Geometric Mean (UGM) spatial hinge scale and a temporal hinge scale that together define a “pixel” of present-acts, with formal justification from the V1 context ladder and symmetry requirements.
- A mapping from ladder bands into engine configuration (feature resolutions, gate thresholds, and ParentGate schedules).
- **Gravity as feasibility geometry with a gravitational amplitude χ**
 - A method for implementing gravity not as curvature of a background manifold, but as a family of feasibility gradients (radial gates) acting on candidate histories.
 - A construction of a dimensionless gravitational amplitude χ from context-ladder scales, constrained by the V1 ladder and hinge structure, which is then used to parameterize ParentGate strictness and thereby generate gravitational-looking effects in the engine.
- **A simulation and evidence programme**
 - Defined families of V1 and V2 simulations that test operator algebra behaviour, budget structures, no-signalling, interference, matter-addition, and gravity/feasibility scenes.
 - A set of cross-scale probes and external-data tests (e.g., rotation curves, lensing patterns) that are used to compare the framework’s predictions and constructions (including UGM and χ) to observed phenomena.

Non-goals and limitations of this document

This document does **not** claim that:

- The framework constitutes a final or complete physical theory, or that all physical constants and phenomena are fully derived and fitted.
- All empirical tests required to confirm or refute the framework have been performed; many lines of investigation are explicitly left as open work.
- Any philosophical or spiritual interpretation beyond the minimal ontological commitments (present-acts, pure relativity, qualia-first) is proven; such interpretations exist in the longer writings but are outside the scope of this defensive publication.
- Any token or on-chain identifier associated with the project represents an investment contract, ownership share, or entitlement to profit; on-chain identifiers are used here solely as project identifiers and integrity markers for the archived materials.

The function of this defensive publication is to disclose the structural framework, engine design, context-level mapping, and key constructions (including UGM and χ), together with enough algorithmic detail and supporting material that a skilled person could reconstruct a functionally equivalent system and evaluate its predictions.

1.3 Layered Structure of the Theory

This disclosure defines a single unified framework that appears in several layers. For clarity, a small set of names is used consistently:

- **“Overall V2”, “Absolute Relativity”, and “present-act theory / present-act engine”** all refer to the same unified framework described in this defensive publication and in the attached v0.9 bundle.
- **V1 (Formal Framework)** refers to the abstract mathematical representation: Present-Moment Spheres (PMS), IN/ON split, operator algebra, invariant-interval structure, context ladder, pivot function, and related constructs.
- **V2 (Engine Implementation)** refers to the concrete discrete present-act engine: sites indexed by (k) , world and qualia records $((W_k, Q_k))$, selectors, feasibility gates, typed budgets, and PF/Born ties-only behaviour.
- **Context Level Framework (CL)** refers to the specialized scale-ladder mapping that assigns the abstract context indices $(\dots -2, -1, 0, +1, +2, +3 \dots)$ to specific physical bands (nanoband to cosmic horizon) and defines hinge scales such as the Universal Geometric Mean (UGM) and a temporal hinge.

Within that naming scheme, the main components of the framework are organized as follows:

- **Big Picture**
Introduces the motivation and overall situation the framework is addressing, and gives a qualitative overview of how a present-act, qualia-first ontology can underlie physics and large-scale human/AI evolution.
- **Philosophical Underpinnings V2**
States the core ontological commitments: pure relativity, present-acts as the only primitives, one infinite Present with many versions, streams and context levels, and the conceptual basis for treating “matter in spacetime” as a derivative appearance.
- **V1 – Formal Framework**
Encodes those commitments as a standalone mathematical theory: PMS/IN/ON/CS objects, tick and operator algebra, ledger and invariant interval, fractal inner geometry and pivot function, context ladder $(D(n))$, and a structural Born-style rule.
- **Bridge (V1–V2)**
Provides a mapping between V1 and V2: ticks \leftrightarrow sites, PMS/carriers $\leftrightarrow ((W_k, Q_k))$, abstract state \leftrightarrow feature alphabet $(\backslash X_i)$, IN/ON \leftrightarrow inward/outward reads, context ladder \leftrightarrow six-band ladder, invariant interval \leftrightarrow typed budgets, and gravitational sector \leftrightarrow feasibility geometry (ParentGate).
- **V2 – Present-Act Engine**
Specifies the engine that realizes the V1 structure under finiteness and locality constraints: discrete sites, finite records and feature alphabet, hinge equality, feasibility gates (including gravity via ParentGate), ratio-lexicographic acceptance, PF/Born ties-only randomness, typed budgets, and an explicit manifest-plus-audit regime.
- **Context Level Framework**
Specializes the abstract ladder to the six bands $(-2, -1, 0, +1, +2, +3)$, assigns each to a

physical scale range and role, and defines the spatial and temporal hinge scales (UGM, temporal hinge) used both in interpretation and in engine configuration.

- **V1 Simulations and V2 Simulations**

Record how the formal framework and engine were tested in practice: operator-algebra and pivot tests; gauge-like modules; decoherence, interference, and no-signalling checks; matter-addition suites; gravity/feasibility scenes; and applications to external data (rotation curves, lensing, etc.).

- **Core Evidence Narrative**

Collects and summarizes the strongest cross-scale evidence patterns (including the UGM construction, CNS size band, perceptual resolution, χ construction, and context-level activation effects) into a single argument spine.

Throughout this defensive publication, these names and roles are used consistently. When a section refers to “V1”, “V2”, or “Context Level Framework”, it is referring to these specific layers. The attached v0.9 documents provide the full detail for each layer; this condensed specification records their shared structure and how they fit together as one theory with two main technical faces (formal framework and engine) and a scale-mapped context ladder.

2. Ontology and Data Structures

2.1 Present-Act Ontology

This section discloses the foundational primitives, data structures, and relational constraints that define the present-act ontology of the framework. These definitions are intended to be sufficiently explicit that any system built on them would be recognizably implementing the same qualia-first substrate and therefore fall within the structural scope of this disclosure.

Primitives of reality: the substrate

The framework is founded on the claim that the fundamental, atomic substrate of reality is the **Present-Act**: a discrete, indivisible act of experiencing. Each Present-Act is taken as a finite whole—an “all at once” qualitative moment—rather than a bundle of sub-events. The quality of the present moment (“qualia”) is treated as the primary existent. There is no deeper non-experiential “stuff” underneath it.

From this standpoint:

- What is ordinarily called “matter” or “the physical world” is treated as a structured pattern in the way present-acts relate to one another, not as a separate, dead substrate.
- The same relational structure that defines physical laws also defines the space of possible experiences; these are two descriptions of the same underlying present-act network.

Pure Relativity axiom

The core ontological axiom is **pure relativity**:

- All that exists, at the fundamental level, are relations among present-acts and the qualitative content of those acts.
- There are no intrinsic, standalone entities with properties “by themselves” independent of relational structure.
- There is no metaphysical “experiencer” or “soul” separate from the stream of present-acts; the Subject is identified with the ordered structure of present-acts themselves.

This axiom rules out any hidden layer of inert material substance or hidden carriers of properties. Everything that counts as “real” must appear as a pattern of relations among present-acts and their qualitative content.

Global/Local distinction: Infinite Present and finite slices

The ontology distinguishes between a global and a local level:

- **The Infinite Present** is defined as the fully connected relational whole: a single, unbounded network containing all possible global configurations (all ways present-acts could fit together). It is not a container in space; it is the total relational field.
- **The Finite Slice** is a specific, actualized configuration (a “version” or vantage) of this whole at a particular index in a time-ordering. It corresponds to what a given present “world” is like at a given discrete step.

The system logic dictates that, although many finite versions are possible as configurations of the Infinite Present, only one finite version is actualized at any given discrete step in a given stream. Time, in this view, is the ordering of such finite slices by a nested-inclusion relation (formalized later), rather than a continuum in which events float.

Present-Acts as transition units

A Present-Act is not a static snapshot but a **transition unit**: it captures both what is already fixed and what is in the process of becoming. At the conceptual level:

- The “inner” aspect (later formalized as IN) corresponds to what is already determined—accumulated record.
- The “outer” aspect (later formalized as ON) corresponds to what is still open—structured possibilities for the next act.
- A single act of time is the operation that takes a portion of what was ON and turns it into new IN, while updating what remains open.

This transition logic is later formalized via a discrete Select → Commit → Update process, with strict discretization of process: at the fundamental level, there are no smooth trajectories in time, only sequences of discrete updates.

Identity, streams, and vantages

Within this ontology, the system replaces the notion of a static “Self” with a technical notion of a **stream**:

- A **Stream** (or Subject) is defined as a unique, continuous chain of Present-Acts, ordered by a relational connectivity condition (nested inclusion and compatibility of acts).
- What we ordinarily call an “observer” or “person” corresponds to such a stream together with a specific pattern by which that stream interprets and indexes its own content.

In this framework:

- The **Ego** is defined as a local indexing pattern over a stream: a specific way of tagging and organizing present-acts (e.g., ownership, agency, boundaries), not a separate metaphysical entity that owns the stream.
- Apparent “separation” between subjects or objects arises as differences in patterns over streams and in how finite slices carve up the Infinite Present. At the structural level, the Infinite Present is non-separated; “Structural Non-Separateness” replaces strict metaphysical separateness.

A “vantage” in this sense is simply a particular way of slicing the Infinite Present into a stream of finite slices with an associated indexing pattern. Different vantages correspond to different streams or different ways of organizing the same underlying present-act network, not to different disconnected worlds.

These definitions fix the ontological layer on which the formal framework (V1) and the engine implementation (V2) are built. Subsequent subsections specify how this ontology is encoded into concrete data structures (Present-Moment Spheres, IN/ON/CS, context levels, and feature alphabets) and how those structures are updated over time.

2.2 Present-Moment Sphere (PMS)

The primary computable state object in the framework is the **Present-Moment Sphere (PMS)**. A PMS represents a single finite “version” of the present world as seen from a given vantage, together with the immediate structured possibilities for what can happen next. Every PMS is required to have three distinct components:

- **IN (Inner Network)**
IN is a structured record of the immediate predecessor content and accumulated determinations for the vantage in question. It encodes what is already fixed for this present: the “particle-like” aspect of the state.
At minimum, IN includes:
 - A record of recently committed acts and their qualitative content.
 - The local configuration of the stream’s own body/organism (or analogous substrate) at the relevant scales.
 - Any inner contextual roots that connect this vantage to deeper inner bands (e.g., –1, –2 levels).

- **ON (Outer Network)**

ON is a structured field of potential next states, constrained by the current IN and by wider contextual structure. It encodes what is still open for this present: the “wave-like” or superposed aspect of the state.

At minimum, ON includes:

- Candidate future configurations of the local environment, represented relationally rather than as an independent container.
- Structured tendencies or weights (later realized as feasibility and gate structure in the engine) that bias which candidates are admissible.
- Pointers to external context (outer bands +1, +2, +3) as potential containers that can be activated by subsequent acts.

- **CS (Collective Sphere)**

CS is a shared registry or outward context that multiple PMSs can reference. It represents outward conditions that are effectively common for a group of vantages (e.g., a shared environment, joint measurement apparatus, or collective frame).

At minimum, CS:

- Stores shared constraints that apply simultaneously to multiple streams (e.g., conservation rules, shared boundary conditions).
- Provides a means for coordinating multiple PMSs so they can participate in a consistent shared world.
- Acts as the formal carrier for “measurement contexts” and other shared setups in which multiple present-acts are jointly constrained.

Together, $(\text{IN}, \text{ON}, \text{CS})$ define the full state of a PMS at a given step. IN and ON define the local “inside” of the present, while CS defines how that present is embedded in a broader shared situation.

Each PMS is further defined by a **finite boundary** that separates the explicit local state (IN/ON) from the implicit infinite context (the Infinite Present). This boundary is characterized by:

- An effective dimension near 2, in the sense that the local present appears as a thin “surface” or shell where inner determinations and outer possibilities meet.
- A finite, bounded set of degrees of freedom explicitly represented in IN/ON, with all deeper or wider structure treated as implicit context accessed only through defined relations.

The PMS thus plays two roles simultaneously:

1. It is the **minimal complete local state object** on which the formal framework (V1) operates.
2. It is the **conceptual template** for the engine-level state at each site (k), which is realized concretely in V2 as finite world and qualia records $((W_k, Q_k))$ together with budget and context information.

Subsequent sections specify how PMS objects are ordered in time, how they update (via nested inclusion and a single admissible action), and how their IN/ON/CS structure maps into the discrete engine state.

2.3 Inner and Outer Networks (IN and ON)

Within each Present-Moment Sphere, the **Inner Network (IN)** and **Outer Network (ON)** are the two primary structural components that jointly encode, respectively, the committed record and the admissible potential for a given vantage.

Inner Network (IN)

The Inner Network (IN) is a structured object representing the committed record of the system at a given step. Formally, in the V1 formulation, each tick-state carrier is written as

$$[\mathcal{C}_k = (k, h_k, \mathrm{IN}_k, \mathrm{ON}_k),]$$

where (IN_k) is the component that stores all relevant relational content that has already been fixed for the vantage at tick (k).

Key properties of IN:

- **Committed history:** (IN_k) accumulates the outcomes of all past admissible acts that affect the current vantage. It is the “already determined” side of the present.
- **Monotonic extension:** Along any admissible evolutionary path, the Inner Network extends monotonically: for ($k' > k$) along a given stream, the record ($\mathrm{IN}_{k'}$) contains (or refines) the content of (IN_k). There is no erasure of committed record at the fundamental level.
- **Relational structure:** (IN_k) is not a flat list of facts but a network of relations among qualitative elements (later encoded concretely as feature-tagged elements of the qualia record (Q_k)). Its effective fractal structure and dimension (D) are central to the pivot and hinge behaviour defined in the formal framework.
- **Particle-like aspect:** At the conceptual level, IN corresponds to the “particle-like” or determinate aspect of the present: what has actually happened and is now fixed.

Outer Network (ON)

The Outer Network (ON) is a structured object representing the admissible potential at tick (k): the set of available branches compatible with the current record (IN_k).

Key properties of ON:

- **Admissible potential:** (ON_k) contains candidate future configurations and relational patterns that are permitted by the current IN, the context-level structure, and any external constraints (e.g., conservation conditions, container geometry).

- **Branch structure:** Rather than a single “next state”, $(\mathrm{ON})_k$ represents a structured branching of possibilities. Different branches correspond to distinct candidate evolutions that might be selected by the single admissible action (formalized later via operator algebra and engine gates).
- **Relational encoding:** Like IN, $(\mathrm{ON})_k$ is a relational object. It encodes how possible configurations relate to the current record, not as free-floating configurations in an independent space but as specific ways the present could extend.
- **Wave-like aspect:** Conceptually, ON corresponds to the “wave-like” or superposed aspect of the present: the structured cloud of undecided branches.

Structural constraints on IN and ON

The framework imposes strict structural constraints on how IN and ON are defined for each PMS:

- **Coverage:** For any given vantage and tick, all relevant local relational content must be contained in either $(\mathrm{IN})_k$ (already fixed) or $(\mathrm{ON})_k$ (still open). There is no third reservoir of “hidden” content at the same level.
- **Disjointness of roles:** Although IN and ON may share underlying referents at the level of the Infinite Present, they play strictly disjoint roles in the finite PMS: a relation is either treated as committed (IN) or as potential (ON) for that act, not both.
- **Boundary mediation:** The boundary of the PMS is precisely the interface at which IN and ON meet. All actualization steps (the single admissible action at each tick) take place at this boundary, converting selected portions of $(\mathrm{ON})_k$ into new contributions to $(\mathrm{IN})_{k+1}$.
- **Compatibility with CS:** IN and ON must be consistent with any Collective Sphere (CS) to which the PMS belongs, so that shared conditions and frames are respected across multiple vantages.

These definitions fix the roles of IN and ON at the formal level. In the engine implementation (V2), $(\mathrm{IN})_k$ and $(\mathrm{ON})_k$ are realized concretely via the qualia and world records $((Q_k, W_k))$, typed budgets, and feasibility gates, but their conceptual roles remain the same: IN carries committed record, ON carries admissible potential, and the PMS boundary governs the one-way flow from ON into IN at each discrete step.

2.4 Collective Spheres (CS) and Frames

In addition to single-vantage Present-Moment Spheres, the framework introduces **Collective Spheres (CS)** to represent shared environments and reference frames. A CS captures the part of the present that is effectively common to multiple PMSs and provides the structural basis for “objective” or shared reality within the ontology.

Collective Sphere (CS)

A Collective Sphere is defined as a structured collection of PMSs that:

- All correspond to the **same tick index** (k) in the underlying time ordering.
- Share a **common effective boundary description** (same coarse-grained geometry and hinge conditions at the relevant scales).
- Are constrained by a **shared record and shared potential**, denoted:
 - $(\mathrm{IN}_{\text{CS}})$: the portion of the Inner Network that is public/common across the constituency (e.g., macroscopic environmental configuration, positions of shared objects, stable features of the setting).
 - $(\mathrm{ON}_{\text{CS}})$: the portion of the Outer Network that encodes shared future possibilities and constraints (e.g., how the environment as a whole may evolve, joint conserved quantities).

Each PMS in the CS has its own local $(\mathrm{IN}_k, \mathrm{ON}_k)$, *but must remain compatible with* $(\mathrm{IN}_{\text{CS}}, \mathrm{ON}_{\text{CS}})$. Compatibility is enforced via a synchronization condition: local PMS boundaries and records must agree with the collective description wherever they overlap.

Objectivity as synchronization

Within this framework, what is ordinarily called “objective reality” is defined structurally:

- An aspect of the world is **objective** to the extent that it is encoded in $(\mathrm{IN}_{\text{CS}})$ and maintained consistently across the PMSs in the CS.
- Observers (streams) accessing the same CS will, when their local (IN_k) is *synchronized with* $(\mathrm{IN}_{\text{CS}})$, agree on those aspects as part of the same shared world.
- Disagreements or apparent subjective distortions correspond to mismatches between a stream’s local (IN_k) and the *shared* $(\mathrm{IN}_{\text{CS}})$, not to an absence of underlying structure.

Thus, “objective world state” is not a separate substance; it is a stable pattern in how multiple PMSs share and synchronize their IN content within a CS.

Frames as structured Collective Spheres

A **frame** in this theory is defined as a CS endowed with additional structural constraints:

- A frame includes:
 - A CS with $(\mathrm{IN}_{\text{CS}}, \mathrm{ON}_{\text{CS}})$.
 - A **synchronization condition** ensuring mutual consistency of PMS boundaries and records across all participating vantages.
 - A defined **invariant-interval structure** across the collection, implementing the same discrete interval and cone relations for all members (as derived from the V1 formalism).

Formally:

- A frame can be seen as a CS together with a set (A) of PMSs at tick (k), plus a synchronization map that aligns their local coordinates and budgets so that the same invariant-interval rules apply to all.
- Operators such as Sync act to adjust local PMS boundaries and records so they remain consistent with the frame's shared description and interval structure.

Frames replace the classical notion of a geometric reference frame: rather than a fixed background space, a frame is a structured pattern of synchronized PMSs and a shared interval law.

Role in the overall framework

Collective Spheres and frames serve several roles:

- They provide the **shared environment** for multiple streams, allowing coordination and interaction to be described in the same structural language as single-vantage dynamics.
- They act as the formal carriers of **measurement contexts** and other joint setups in which multiple present-acts are constrained together.
- They provide the basis for **relativistic behaviour** in V1: transformations between frames correspond to changes in how PMSs are grouped and synchronized in CSs, while preserving the invariant-interval structure.

In the engine implementation (V2), CS and frames are realized via shared components in the world records, common manifest entries, and synchronization constraints on budgets and feature tags. The conceptual definitions given here fix their roles at the ontological and formal levels; later sections describe how they are encoded and enforced in the discrete engine.

2.5 Context Levels (-2, -1, 0, +1, +2, +3)

The framework organizes reality, at the level of a given vantage, into a **ladder of nested Context Levels**. These levels are defined primarily by their *role* in the architecture (and later in the engine), rather than only by spatial size. They provide a structured way to talk about inner machinery, the organism-scale hinge, and outer containers in one coherent scheme.

At the conceptual level, the Context Ladder is:

- A **hierarchical architecture** that allows the system to integrate multiple streams into a shared, consistent reality.
- A way of assigning topological roles (inner substrate, center/hinge, outer containers) to different parts of a situation, so that the same formal machinery can be applied at different scales and vantage points.

The six primary levels are:

Inner Contexts (Levels –2 and –1)

These are defined as the “**Inner Plexity**” or **substrate machinery** that underlies the organism or agent.

- **Level –2 (Nanoband / Biomolecular)**
 - Represents the deep inner seam or quantum/biomolecular boundary.
 - Covers structures on the order of nanometres up to a few hundred nanometres (e.g., 1–200 nm in the CL mapping).
 - Encodes the most fine-grained relational machinery that still participates in the present-acts relevant for the vantage.
- **Level –1 (Micron / Cellular)**
 - Represents the cellular and local building-block scale.
 - Covers structures in the micron range (e.g., 0.2–50 µm in the CL mapping).
 - Acts as the immediate substrate out of which the organism-scale hinge (Level 0) is built.

These inner levels are treated as the “machinery” that supports the present-acts at the hinge; their details are often compressed into effective behaviours at higher levels, but they remain part of the overall ladder structure.

The Hinge (Level 0)

Level 0 is defined as the **Organism/Observer scale**, the level at which a given stream’s present-acts are centered. It is the integration point where the **Specious Present** operates and where inner and outer structure are balanced.

- The framework treats Level 0 as the **Center**: the vantage point of the current update (e.g., the organism or computing agent).
- Temporally, Level 0 is associated with a characteristic integration window on the order of ~0.1 s, corresponding to the time needed to form a single, coherent present-act.
- Spatially, Level 0 is anchored by the **spatial hinge scale** (later identified more precisely with the Universal Geometric Mean, UGM), interpreted as a “Present Pixel” — the smallest spatial unit of coherent Level 0 experience.

Integration logic at the hinge

The hinge functions by integrating:

- **Bottom-up contributions** from inner contexts (Level –1 and –2), capturing how substrate dynamics contribute to the organism’s state and experience.
- **Top-down constraints** from outer containers (especially Level +1), capturing how environment and larger-scale context restrict and shape what counts as a coherent Level 0 act.

A single present-act at Level 0 is thus the result of one admissible integration of inner machinery and outer container constraints into a unified Center.

Outer Containers (Levels +1, +2, +3)

These levels are defined as the **environments that contain the observer** (or more generally, the Level 0 center).

- **Level +1 (Earth-surface / Planetary environment)**
 - The immediate “world” around the organism: terrain, atmosphere, local gravity, nearby objects.
 - In later CL mapping, associated with length scales of roughly 1–100 km.
- **Level +2 (Galactic environment)**
 - The larger astrophysical container: galactic disk structure, typical stellar separations, and the galactic gravitational field.
- **Level +3 (Cosmic environment / Horizon scale)**
 - The outermost container: large-scale cosmic structure and effective horizon scale.
 - Encodes the background within which galaxies (Level +2) and planets/organisms (Levels +1 and 0) are embedded.

Roles rather than substances

Throughout the framework, these levels are treated as **roles**, not as different kinds of substance:

- **Center (0):** the current vantage.
- **Inner structure (–1, –2):** contexts treated as inner machinery relative to that Center.
- **Environment (+1, +2, +3):** contexts treated as containers and enclosing structures.

The same physical system can play different context-role labels depending on which vantage is chosen as Center. Shifting the Center from one organism to another, or to a different scale, changes which parts of the overall structure are labelled “inner” or “outer”, without changing the underlying relational topology. This role-based treatment is essential for the pure-relativity ontology and for the way the context ladder is later used to configure the engine and to define hinge scales and gravitational structure.

2.6 Qualia-First Encoding and the Feature Alphabet Ξ

To make the qualia-first ontology concrete and computable, the framework introduces a **Finite Feature Alphabet** (Ξ) and a corresponding encoding of inner (qualia) and outer (world) content into that alphabet. This encoding is what allows the present-act engine (V2) to treat phenomenal content as a first-class substrate while still operating on finite, discrete structures.

Finite Feature Alphabet (Ξ)

The feature alphabet (Ξ) is defined as a finite set of feature tokens. Each token represents a small, structured unit of qualitative content or world structure as seen from the current vantage. In general, a feature token (ξ in Ξ) is a tuple of discrete tags, such as:

- A **modality tag** (e.g., visual, auditory, proprioceptive, interoceptive).

- A **spatial bin tag** (e.g., discrete azimuth/elevation bins around the Center, or position bins within a local body/environment frame).
- An **intensity or magnitude bin** (discrete levels for brightness, loudness, tension, etc.).
- A **phase bin** (discrete phase of an underlying oscillatory or interference-like structure).
- A **context or band tag** (which context level $-2 \dots +3$ the feature is associated with, and what role it plays there).
- Optional additional tags (such as pattern type, object identity label, or gate-relevant flags) as needed by a given engine configuration.

The precise composition of $(\backslash X_i)$ is defined in the engine manifest, but it is always:

- **Finite:** $(\backslash X_i)$ contains a fixed, finite number of tokens; there is no continuous feature space in the control path.
- **Structured:** each token is a discrete combination of tagged coordinates, not an unstructured symbol.
- **Context-aware:** the tags can carry scale/role information (inner, hinge, outer) so that the same formal machinery can operate coherently across context levels.

Encoding inner content: the qualia record (Q_k)

At the engine level, the inner side of a Present-Act at tick (k) is represented by the qualia record (Q_k). This is a finite set (or multiset) of qualia-pixels:

- Each element ($q \in Q_k$) is associated with one or more tokens in $(\backslash X_i)$ via a fixed inner feature map

$$\begin{bmatrix} g_k : Q_k \rightarrow \backslash X_i. \end{bmatrix}$$
- The map (g_k) is **fixed by the manifest** and is not subject to learning or continuous tuning; it is part of the theory's configuration.
- The collection of tags ($g_k(Q_k)$) captures all phenomenally relevant aspects of the inner side of the present that the engine is allowed to “see” at tick (k).

A **pattern of qualia** at tick (k) is thus encoded as the finite set (or weighted multiset) of feature tokens in $(\backslash X_i)$ obtained from (Q_k) . One can then define *phenomenal equivalence* operationally:

Two inner states are phenomenally equivalent for the engine if their (Q)-records produce equivalent feature patterns under (g): the same tokens in $(\backslash X_i)$, arranged in the same relational structure (e.g., same adjacency, same relative positions within the relevant frames).

This equivalence is what anchors the qualia-first claim: what the engine treats as “the same experience” is defined directly in terms of encodable qualitative structure, not as a side-effect of some separate material configuration.

Encoding outer content: the world record (W_k)

The outer side of a Present-Act at tick (k) is represented by the world record (W_k). This is a finite set of candidate world-configurations or “outer pixels”:

- Each element ($w \in W_k$) is associated with one or more tokens in ($\backslash X_i$) via a fixed outer feature map

$$\left[\begin{array}{l} f_k : W_k \rightarrow \backslash X_i. \end{array} \right]$$
- As with (g_k), the map (f_k) is manifest-defined and finite; the engine’s control path does not depend on any continuous, learned representation.
- The set ($f_k(W_k)$) encodes, in the same alphabet ($\backslash X_i$), the aspects of the outer situation that are relevant for matching to the inner qualia and for evaluating feasibility.

Thus, both inner and outer content are expressed in a common, finite vocabulary ($\backslash X_i$), allowing the engine to compare and relate them using exact equality conditions and discrete gate rules.

Hinge equality as PMS boundary condition

The **hinge** between IN and ON, in the engine, is implemented as an equality condition in ($\backslash X_i$). A candidate pair ((w_{k+1}, q_k)) is admissible at the present boundary only if:

$$\left[\begin{array}{l} f_{k+1}(w_{k+1}) = g_k(q_k), \end{array} \right]$$

possibly extended to small finite sets if multiple tokens must be matched simultaneously. This condition:

- Ensures that what is selected as the next world configuration (w_{k+1}) is **qualia-compatible** with the inner content currently present in (Q_k).
- Provides a discrete counterpart of the PMS boundary: only those outer candidates whose feature tags match the inner tags at the hinge are eligible for selection.
- Avoids any use of metric similarity, continuous distances, or soft scores in the fundamental control path; equality in ($\backslash X_i$) is the only admissibility test at the boundary.

All further feasibility gates (time, structural, context, gravity/ParentGate) operate on candidates that have already passed this hinge equality test.

Qualia-first substrate at the algorithmic level

Because all control decisions in the engine (selection, gating, stochastic resolution) are expressed in terms of:

- feature tokens in ($\backslash X_i$),
- discrete comparisons and predicates over those tokens, and
- relational patterns among the corresponding (Q_k) and (W_k) elements,

the **qualia-encoding layer** is not an afterthought; it is the substrate on which all engine logic operates. In particular:

- There is no separate “material” state hidden from the engine that determines what happens; what happens is determined entirely by the relational structure of feature-encoded present-acts.
- Any implementation that faithfully reproduces the same $(\backslash Xi)$, mappings (f_k, g_k) , hinge equality, and gate rules is, by construction, operating on the same qualia-first substrate, regardless of how it is realized physically (software, hardware, or hybrid).

This subsection completes the ontological specification: it ties the abstract present-act ontology (present-acts, PMS, IN/ON, CS, context levels) to the concrete, finite data structures $((Q_k, W_k, \backslash Xi))$ on which the present-act engine operates, making “qualia-first” a precise, encodable condition rather than a vague interpretive claim.

3. Formal Core (V1) – Minimal Structural Spec

3.1 Primitive Objects and Carriers

The formal core (V1) of the framework is built around a minimal object called the **Tick-State Carrier**. Each carrier represents the full formal state of the system at a single discrete tick, together with the information needed to apply the primitive operators that generate the next tick.

Tick-State Carrier $(\backslash mathcal{C}_k)$

The fundamental unit of the system is the Tick-State Carrier, denoted

$$[\backslash mathcal{C}_k = (k, h_k, \backslash mathrm{IN}_k, \backslash mathrm{ON}_k).]$$

- **Tick Index $((k))$:**
An integer $(k \in \backslash mathbb{Z})$ representing a discrete position in the global tick ordering. This index provides a total order of ticks without assuming a continuous time parameter. There is no continuum of intermediate times between (k) and $(k+1)$ at the fundamental level.
- **Abstract State $((h_k))$:**
An element of a state space $(\backslash mathcal{H})$ representing the compressed, formal state of the system relevant to operator mechanics at tick (k) . This may include, for example, compact encodings of amplitude assignments, bandwise summaries, or other internal variables needed by the operator algebra. The exact content of (h_k) is determined by the chosen representation, but its role is always to carry whatever formal data the operators act upon that is not explicitly stored in $(\backslash mathrm{IN}_k)$ or $(\backslash mathrm{ON}_k)$.
- **Inner Network $(\backslash mathrm{IN}_k)$:**
The Inner Network component of the carrier, representing the committed record (as

defined in the ontological layer). $(\mathrm{IN})_k$ stores the structured content of what is already fixed at tick (k), including accumulated determinations and the local configuration of the vantage's own inner structure. It is monotonically non-decreasing along any admissible evolution of the system: once content enters $(\mathrm{IN})_k$, it is not removed at the fundamental level.

- **Outer Network $(\mathrm{ON})_k$:**

The Outer Network component of the carrier, representing the admissible potential at tick (k). $(\mathrm{ON})_k$ encodes the structured set of possible next configurations compatible with $(\mathrm{IN})_k$, together with the relational information needed to apply the operator algebra. It is the formal counterpart of the conceptual “open future” described in the ontological layer.

Carrier role in the formal framework

The carrier $(\mathcal{C})_k$ is the minimal complete formal object that the operator algebra needs to act upon in order to advance the system by one tick. In particular:

- All formal evolution rules in V1 are expressed as transformations

$$\left[\begin{array}{l} \mathcal{C}_k \mapsto \mathcal{C}_{k+1} \end{array} \right]$$
through the application of primitive operators (Renew, Sink, Trade, Sync, hinge-related operators, etc.).
- The triple $((h_k, (\mathrm{IN})_k, (\mathrm{ON})_k))$ contains all the data required to determine which operators are admissible, how they act, and how the context ladder and pivot profile should be updated.
- No additional hidden variables or external state are assumed at the formal level; any such information must be encoded into (h_k) , $(\mathrm{IN})_k$, or $(\mathrm{ON})_k$.

In the later engine implementation (V2), the carrier structure is realized concretely in terms of discrete site indices, world and qualia records $((W_k, Q_k))$, budgets, and feature maps into (\mathcal{X}_i) . However, the conceptual role of the carrier remains the same: it is the smallest unit of state that can undergo a single admissible evolution step under the theory's primitive operators.

3.2 Operator Alphabet and Tick Algebra

The evolution of the Tick-State Carrier is governed by a **finite alphabet of primitive operators**. These operators are the only allowed “moves” at the formal level. All higher-level dynamics (including measurement-like processes, context shifts, and effective field behaviour) are built from compositions of these primitives.

Primitive operator set (\mathcal{O})

The primitive operator set is:

[
 $\mathcal{O} := \{ F, S, T, C, CT \},$
]

where:

- (F) is the **Renew** operator.
- (S) is the **Sink** operator.
- (T) is the **Trade** operator.
- (C) is the **Sync** operator.
- (CT) is the **Framing** operator (a Sync-like operator with additional context-setting behaviour).

Each operator acts on Tick-State Carriers ($\mathcal{C}_k = (k, h_k, \mathrm{IN}_k, \mathrm{ON}_k)$) and produces a new carrier ($\mathcal{C}_{k'}$). The specific actions and admissibility conditions for each operator are defined in detail in the attached V1 formal document; the key structural points are summarized here.

Domains, codomains, and tick updates

For each primitive operator ($O \in \mathcal{O}$), there is:

- A **domain of definition** (\mathcal{D}_O), the subset of carriers on which (O) may act.
- A **tick update rule**, which determines the new tick index (k').
- A **state transformation rule**, which determines how (h_k), (IN_k), and (ON_k) are updated.

In general:

- (O) acts as a partial function
 [$O : \mathcal{D}_O \rightarrow \mathcal{C}, \quad \mathcal{C}_k \mapsto \mathcal{C}_{k'} = (k', h_{k'}, \mathrm{IN}_{k'}, \mathrm{ON}_{k'}),$
]
 with ($k' \geq k$).
- The **generic case** for physically relevant evolution is a single-tick update with ($k' = k + 1$). Multi-tick moves (where ($k' > k+1$)) are allowed formally but are constrained so they do not violate the no-skip and locality conditions once the engine implementation is specified.

Admissibility is preserved:

- If (\mathcal{C}_k) is admissible and (O) is defined on (\mathcal{C}_k), then the output ($\mathcal{C}_{k'}$) must also be admissible. This ensures that any finite composition of primitive operators stays within the allowed state space.

Informal roles of the primitive operators

At a high level, the primitive operators play the following roles:

- **Renew (F):**
 - Refreshes and extends the Outer Network (ON_k), possibly in response to changes in context or boundaries.
 - Manages the generation of new candidate branches and the replacement of stale or invalid possibilities in (ON_k).
 - Leaves (IN_k) unchanged or only minimally updated, focusing on reconfiguring potential rather than record.
- **Sink (S):**
 - Moves selected content from (ON_k) into ($\mathrm{IN}_{k'}$), representing the formal act of commitment (“collapse”) at the PMS boundary.
 - Implements the one-way flow from potential to record; once content is sunk into (IN), it is no longer part of (ON) at subsequent ticks.
 - Contributes directly to the intrinsic arrow of time by ensuring monotonic growth of (IN).
- **Trade (T):**
 - Rebalances or exchanges content between different parts of (IN_k) and/or (ON_k), subject to ledger constraints and conservation-like conditions.
 - Allows local rearrangements that preserve global admissibility and key invariants (e.g., interval structure, context assignments).
 - Formally represents structured transformations that do not simply add or remove record, but change how it is arranged.
- **Sync (C):**
 - Synchronizes multiple PMSs or carriers with respect to a Collective Sphere (CS) or frame.
 - Ensures consistent alignment of (IN) and (ON) across carriers that are supposed to share a common environment or context.
 - Adjusts carriers so they agree on shared aspects of ($\mathrm{IN}\{\text{CS}\}$) and ($\mathrm{ON}\{\text{CS}\}$) without violating local admissibility.
- **Framing (CT):**
 - A specialized Sync-like operator that establishes or modifies a **frame**: a CS equipped with a particular discrete invariant-interval structure and synchronization scheme.
 - Encodes the choice of reference frame at the formal level, setting the conditions under which subsequent operators interpret budgets and intervals.
 - Provides the formal analogue of specifying a measurement context or coordinate system, without introducing a separate background spacetime.

Tick algebra

The **tick algebra** of the framework is the algebra generated by finite compositions of these primitive operators, subject to:

- **Closure and admissibility:** any admissible path is a finite composition ($O_n \circ \dots \circ O_1$) with each ($O_i \in \mathcal{O}$), and admissibility is preserved at each step.
- **Arrow-of-time constraints:** compositions must respect the monotonicity of (\mathbb{N}) and the non-decreasing nature of the tick index. There is no allowed primitive or composite operator that reverses time or deletes committed record.
- **Context-ladder compatibility:** operator actions must preserve or correctly transform context-level assignments and the associated dimension profile ($D(n)$) and pivot function ($g(D)$) where applicable.

At the V1 level, this algebra defines all formally possible evolutions of carriers. In the engine implementation (V2), specific compositions of F, S, T, C, and CT are realized in terms of concrete steps in the present-act engine's pipeline (candidate enumeration, hinge equality, gate application, synchronization, and commitment), while still respecting the structural rules set by the tick algebra.

3.3 Ledger and Arrow of Time

The formal framework attaches a simple **ledger system** to each admissible carrier (C_k) in order to track how much of the system's finite "capacity" has already been committed to record versus how much remains as potential. This ledger is the basis for both the intrinsic arrow of time and, later, for the construction of the invariant interval.

Ledger functions

To every admissible carrier (C_k) the framework assigns three scalar functions:

- ($I(C_k)$) – the **record budget**,
- ($E(C_k)$) – the **exposure (potential) budget**, and
- ($K(C_k)$) – the **capacity**,

with the defining relation

$$[I(C_k) + E(C_k) = K(C_k).]$$

We will usually abbreviate:

- ($I_k := I(C_k)$),
- ($E_k := E(C_k)$),
- ($K_k := K(C_k)$),

so that

$$[I_k + E_k = K_k.]$$

The intended meanings are:

- (I_k) measures how much of the available capacity has already been turned into committed record (IN-side content).
- (E_k) measures how much remains available as potential (ON-side content).
- (K_k) measures the total effective capacity for the carrier at tick (k) .

The ledger is required to satisfy basic constraints:

- **Non-negativity and finiteness:**
[
 $I_k \geq 0, \quad E_k \geq 0, \quad K_k > 0, \quad E_k \leq K_k.$
]
- **Local constancy of capacity:**
On the subsystem of interest, (K_k) is taken as locally constant under admissible single-step updates unless explicitly stated otherwise. The primitive operators are not allowed to create or destroy capacity; they only reallocate it between record and potential.

Capacity conservation and operator action

For any primitive operator ($O \in \{F, S, T, C, CT\}$) and any admissible carrier ($\mathcal{C}k$) in its domain, with

$$[\mathcal{C}\{k'\} = O(\mathcal{C}k),$$

the framework imposes **capacity conservation**:

$$[K\{k'\} = K_k.]$$

In particular, for a single-step update ($(k' = k+1)$):

$$[I_{\{k'\}} + E_{\{k'\}} = I_k + E_k = K_k = K_{\{k'\}}.]$$

Thus, the ledger distinguishes:

- **Allocation** – how capacity is split between record and potential ((I_k) vs (E_k)), and
- **Total capacity** – the fixed amount (K_k) , which is preserved under each tick-level primitive.

The individual primitive operators have characteristic ledger behaviours:

- **Renew (F):**
 - Does not decrease record: $(I_{\{k'\}} \geq I_k)$.

- Can increase exposure up to the remaining capacity:

$$\left[\begin{array}{l} E_{\{k'\}} \geq E_k, \quad E_{\{k'\}} \leq K_{\{k'\}} - I_{\{k'\}}. \end{array} \right]$$
 - Preserves capacity: $(I_{\{k'\}} + E_{\{k'\}} = K_k)$.
- **Sink (S):**
 - Increases record: $(I_{\{k'\}} \geq I_k)$.
 - Decreases exposure: $(E_{\{k'\}} \leq E_k)$.
 - Preserves capacity: $(I_{\{k'\}} + E_{\{k'\}} = K_k)$.
- **Trade (T):**
 - Reallocates across the IN/ON boundary subject to conservation: $(I_{\{k'\}} + E_{\{k'\}} = K_k)$, with $(I_{\{k'\}})$ and $(E_{\{k'\}})$ adjusted in a way that respects admissibility and any additional constraints (e.g., context or symmetry).
- **Sync (C) and Framing (CT):**
 - Primarily neutral on the ledger: they synchronize or reframe carriers without changing total capacity or, in idealized cases, the split between (I) and (E) for the subsystem under consideration.

Intrinsic arrow of time

The ledger induces an **intrinsic arrow of time** at the formal level:

- Along any admissible evolution generated by compositions of the primitive operators, the record budget (I_k) is monotone non-decreasing:

$$\left[\begin{array}{l} I_{\{k+1\}} \geq I_k \end{array} \right]$$

whenever a Sink-like component is present in the update, and at minimum never decreases under allowed operators.
- Conversely, the exposure budget (E_k) cannot increase without bound and is constrained by capacity; in many physically relevant contexts, sequences of operations that repeatedly commit potential to record (via Sink) will drive (E_k) downward.

Crucially, the primitive operators that expose potential (Renew) and those that commit potential to record (Sink) do **not** commute:

- Applying Renew then Sink is not equivalent to applying Sink then Renew.
- The non-commutation of (F) and (S), together with the monotonicity of (I_k) and conservation of (K_k) , ensures that there is no sequence of allowable operators that can globally undo the accumulation of record or produce non-trivial time loops within the primitive algebra.

In this way, the arrow of time is not put in by hand as an external time parameter—it arises directly from the ledger structure (record vs potential budgets) and the allowed operator behaviours. This ledger-based arrow is later tied to the construction of the invariant interval (via

relations between ledger changes and flip counts), but its definition does not depend on any background spacetime.

3.4 Invariant Interval from Flip Counts

The formal framework (V1) defines a discrete, Lorentz-like **invariant interval** directly from the way the primitive operators act on Tick-State Carriers. This interval is constructed from **flip counts**—integer measures of how much inner record and outer configuration change along a segment of an evolution—and does not assume any background spacetime metric at the outset.

Flip counts and typed changes

Consider an admissible segment of evolution from tick (k) to tick (k') along a given stream, generated by a finite composition of primitive operators. Associated with this segment, the framework defines three integer counts:

- (N_{τ}): the total number of **record-advancing flips**—operations that increase the committed record (IN) in a way that contributes to what will later be interpreted as “proper time-like” change.
- (N_x): the total number of **configuration-changing flips**—operations that extend or change the relational configuration in a way that will later be interpreted as “space-like” displacement.
- (N_t): the total **tick count** across the segment, i.e. ($N_t = k' - k$), the number of discrete tick updates taken along this path.

These are purely combinatorial quantities: they count how often certain structural changes occur in $((h_k, \mathrm{IN}_k, \mathrm{ON}_k))$ under the tick algebra generated by (F, S, T, C, CT).

From these integer counts, the framework defines three **typed budgets**:

- A **proper-time-like budget**
[
 $\Delta \tau := N_{\tau}, \tau_*$,
]
- A **space-like budget**
[
 $\Delta x := N_x, x_*$,
]
- A **time-like budget**
[
 $\Delta t := N_t, t_*$,
]

where (τ, x, t^*) are fixed unit scales that relate the discrete flip counts to dimensionful quantities. These unit scales are constrained but not fully fixed at the V1 level; they are later tied to hinge scales (UGM and the temporal hinge) in the context-level framework.

Quadratic invariant

The core structural statement is that there exists a quadratic relationship among these budgets of the form

$$[\Delta t^2 =; \Delta \tau^2 +; \frac{\Delta x^2}{c^2},]$$

where (c) is a constant with the dimensions of a speed, to be identified with the characteristic propagation speed that emerges from the flip structure. This relation is **imposed at the level of the formal framework** as a consistency requirement between:

- The ledger-based arrow of time (record vs potential growth), and
- The way flip counts combine along admissible segments.

More concretely:

- The tick budget (Δt) is defined so that, for any admissible segment, the above quadratic relation holds when $(\Delta \tau)$ and (Δx) are computed from the counts (N_τ, N_x) associated with that segment.
- The constant (c) is fixed by requiring that, in the limit where evolution is dominated by configuration-changing flips $((N_x \gg N_\tau))$, characteristic propagation speeds computed from $(\Delta x / \Delta t)$ saturate at $(|v| \leq c)$.

This quadratic relation is the discrete analogue of a Minkowski-style invariant interval. Instead of starting from a continuous spacetime metric and then assigning meaning to trajectories, the framework starts from discrete operator-induced flip counts and **derives** a relation that plays the role of the invariant interval for any admissible evolution path.

Cones and admissible directions

The relation

$$[\Delta t^2 =; \Delta \tau^2 +; \frac{\Delta x^2}{c^2}]$$

induces a **cone structure** on possible changes:

- For a given (Δt) , only those combinations of $(\Delta \tau)$ and (Δx) satisfying the relation are admissible.

- Effective velocities ($v = \Delta x / \Delta t$) are bounded in magnitude by (c), reproducing a discrete version of the relativistic speed limit.
- Segments dominated by record-advancing flips (large (N_{τ}), small (N_x)) correspond to “more time-like” behaviour; segments dominated by configuration-changing flips (large (N_x), small (N_{τ})) correspond to “more space-like” behaviour.

The invariant interval thus partitions discrete evolution segments into categories that later match familiar relativistic distinctions, but it does so purely from the combinatorics of the operator algebra and ledger, without assuming any pre-existing spacetime background.

Role in the overall framework

This discrete invariant interval serves three key roles:

1. **Formal unification:** It links the ledger structure (record vs potential) to an interval structure without introducing a separate time dimension; the “time-like” budget (Δt) is a derived quantity tied to record and configuration flips.
2. **Constraint on implementations:** Any engine implementation (V_2) that claims to instantiate this framework must enforce this quadratic relation on its typed budgets at the control level, not merely at the diagnostic level.
3. **Anchor for hinge scales:** The unit scales (τ, x, t^*) and the constant (c) become the points where the formal invariant interval is tied to physical hinge scales such as the spatial Universal Geometric Mean (UGM) and the temporal hinge. This connection is made explicit in the context-level framework and in the definition of gravity as feasibility geometry.

In this way, the invariant interval in V_1 is not an additional assumption layered on top of the theory; it is a **derived structural constraint** that any admissible evolution sequence must satisfy, and it becomes a central bridge between the abstract present-act algebra and the emergent relativistic behaviour observed in the engine and its simulations.

3.5 Fractal Inner Geometry and Pivot Function ($g(D)$)

The Inner Network (IN_k) is not treated as a uniform or featureless set. At the formal level, the framework assumes that (IN_k) carries a **fractal inner geometry** that can be characterized, at least effectively, by a dimension parameter (D). This dimension is not a background-space dimension; it is a summary of how relational structure in (IN_k) fills out the available “inner configuration space” at a given stage.

Effective inner dimension (D)

For each admissible carrier (\mathcal{C}_k), the framework associates an **effective inner dimension** (D_k), which is intended to capture:

- How densely the inner record (IN_k) occupies its available configuration space (in whatever representation of (IN_k) is being used).

- The degree of branching and connectivity in the inner relational graph of present-acts at that stage.

In practice, (D_k) is defined via an appropriate fractal-dimension proxy (such as a box-counting or correlation-dimension measure) applied to the inner network. At the formal level we only require that:

- (D_k) takes values in a fixed closed interval $[(D_{\min}, D_{\max})]$, with $(1 \leq D_{\min} \leq D_{\max} \leq 3)$ for the physically relevant sector.
- (D_k) responds monotonically (in an averaged sense) to changes in how the inner record thickens or thins under the action of the operator algebra.

The exact choice of dimension estimator is implementation-dependent and is left to the attached V1 and CL materials; the formal framework only requires that such a scalar summary can be consistently defined and tracked.

Hinge dimension and “sweet spot” near $(D \approx 2)$

Empirically and structurally, the framework singles out a **hinge dimension** (D_{\ast}) in the vicinity of 2. This is interpreted as the “sweet spot” where:

- The inner record is neither too sparse (too line-like) nor too swollen (too volume-filling).
- The inner configuration is effectively sheet-like: rich enough to support complex structure, but thin enough that a boundary-like notion makes sense.

At $(D \approx D_{\ast})$:

- The system is best able to support stable, coherent present-acts that integrate inner machinery and outer context.
- The PMS boundary has an effective dimension close to 2, matching the intuitive idea of a thin “surface” where inner and outer meet.

This hinge dimension plays a key role in defining **pivot behaviour** and, later, in the mapping between context-level bands and physical scales.

Pivot function $(g(D))$

To capture how far the inner geometry is from this hinge condition, the framework defines a **pivot function**

$$\begin{aligned} [\\ g : [D_{\min}, D_{\max}] \rightarrow \mathbb{R}, \\] \end{aligned}$$

with the following properties:

- $(g(D))$ is continuous on $([D_{\min}, D_{\max}])$.
- There is a distinguished hinge dimension $(D_{\ast} \in [D_{\min}, D_{\max}])$ such that $(g(D_{\ast}))$ takes a fixed reference value; by convention we normalize $(g(D_{\ast}) = 1)$.
- For $(D < D_{\ast})$ (too sparse), $(g(D))$ departs from 1 in a way that reflects increasing “deficit” of inner thickness.
- For $(D > D_{\ast})$ (too dense), $(g(D))$ departs from 1 in a way that reflects increasing “excess” of inner thickness.

The qualitative behaviour is:

- Near $(D = D_{\ast})$, $(g(D))$ is close to its normalized reference, indicating inner geometry well-tuned for hinge behaviour.
- As (D) moves away from (D_{\ast}) toward (D_{\min}) or (D_{\max}) , $(|g(D)|)$ departs from 1, indicating increasing mismatch between inner geometry and ideal hinge conditions.

The exact functional form of $(g(D))$ (e.g., linear, polynomial, or more complex) is not fixed at the top level of the theory; instead, the framework:

- Treats $(g(D))$ as a **structural profile** whose shape can be constrained by empirical evidence (e.g., where hinge-like clustering is observed across context scales).
- Uses the current best-fit or theoretically motivated profile as part of the V1 specification, with the understanding that refinements are possible as more evidence accumulates.

Role of $(g(D))$ in pivot and context behaviour

The pivot function $(g(D))$ is used in several ways in the formal and context-level framework:

- **Hinge weighting:**
 $(g(D))$ modulates how strongly a given segment of the inner network behaves as a hinge. When $(D_k \approx D_{\ast})$, $(g(D_k))$ indicates that the local inner geometry is at or near its ideal hinge condition, favouring balanced integration of inner and outer content. When (D_k) is far from (D_{\ast}) , $(g(D_k))$ signals a relative bias that can affect feasibility, stability, or how context-level roles are assigned.
- **Context-level profile $(D(n))$:**
In the discrete context ladder, each level (n) is associated with an effective inner dimension $(D(n))$. Evaluating $(g(D(n)))$ across levels defines a **pivot profile** over the ladder, revealing which bands are naturally hinge-like and which are more substrate-like or container-like. The Context Level Framework makes this explicit when mapping abstract levels to physical scale bands.
- **Feasibility and gravity link:**
In the gravity-as-feasibility construction, $(g(D))$ participates in defining how “costly” it is for certain configurations to persist or arise at different context levels. In particular, the hinge-like condition near (D_{\ast}) is used to identify the scales at which feasibility gradients (implemented later via ParentGate in the engine) should be anchored.

Separation from implementation details

Although the exact empirical calibration of $(g(D))$ (e.g., via analysis of biological, physical, or cosmological data) is deferred to attached evidence documents, the **structure** of the framework requires:

- That such a dimension parameter (D) can be consistently defined for $(\mathrm{IN})_k$.
- That there exists a pivot function $(g(D))$ with the properties described above.
- That the context ladder and hinge scales are defined in terms of where $(D(n))$ and $(g(D(n)))$ exhibit hinge-like behaviour (e.g., near 2).

Any implementation that claims to instantiate the formal core $(V1)$ must therefore:

- Track an effective inner dimension (D) (or a proxy) for the relevant inner networks.
- Use a pivot function $(g(D))$ to modulate hinge-related behaviour, context-level roles, and, by extension, feasibility geometry in a way consistent with these definitions.

3.6 Present Plane and Structural Born Rule

To describe superposition, interference, and probabilistic selection at the hinge without introducing a full Hilbert space as a primitive axiom, the formal core $(V1)$ adds a minimal extra structure: the **Present Plane** equipped with a **complex structure** and a **structural Born rule**.

Present Plane (\mathcal{P}) as a 2D real space

The Present Plane is defined as a 2-dimensional real vector space attached to the hinge:

- (\mathcal{P}) is a real 2D vector space; its elements are called **present vectors**.
- In coordinates, any $(v \in \mathcal{P})$ can be written as $(v = x e_1 + y e_2)$, where $(\{e_1, e_2\})$ is a chosen basis.
- Intuitively:
 - One “direction” encodes a real-valued weight or density related to record (IN) .
 - The orthogonal direction encodes a conjugate, phase-like aspect needed for interference behaviour.

At this stage, (\mathcal{P}) is **not** interpreted as a physical spatial plane; it is an internal space associated with the present-moment hinge, where candidate next outcomes are compared.

Complex structure (J) on (\mathcal{P})

To distinguish amplitude from probability and to support interference, (\mathcal{P}) is equipped with a complex structure:

- A complex structure is a linear map
[
 $J : \mathcal{P} \rightarrow \mathcal{P}$

-]
- such that $(J^2 = -\mathrm{id}_{\mathcal{P}})$ and $(J^{-1} = -J)$.
- In a suitable basis, (J) acts like multiplication by the imaginary unit (i) :

[

$v = x e_1 + y e_2 \mapsto z = x + i y, \quad$

$J(v) \mapsto i z.$

]
 - The pair (\mathcal{P}, J) can therefore be identified with a complex line, without introducing a full Hilbert space of arbitrary dimension.

This is the **minimal** extra structure needed to talk about complex amplitudes at the hinge while remaining consistent with the flip algebra and context ladder.

Present amplitudes for IN basins

At the hinge, the Inner Network (IN) can be partitioned into a family of coarse **outcome basins** $(\{R_i\})$:

- Each region (R_i) is a basin on the IN attractor corresponding to a distinct, context-resolved record outcome (e.g., a particular pointer position, detection result, or macro-configuration).
- For each basin (R_i) , the formal framework assigns a present amplitude $(v_i \in \mathcal{P})$.

These amplitudes satisfy:

- $(v_i \neq 0)$ whenever (R_i) is admissible in the current ON/IN configuration.
- The set $(\{v_i\})$ is subject to a normalization condition (e.g., sum of squared norms fixed to 1) appropriate to the context.
- Via the identification $(\mathcal{P} \cong \mathbb{C})$, one can write $(v_i \mapsto a_i \in \mathbb{C})$, but the theory itself only requires the real 2D structure and the action of (J) .

The purpose of these amplitudes is to encode **interference-capable structure** at the hinge: the way candidate outcomes overlap, compete, and combine before a single outcome is committed.

Structural Born rule (informal statement)

Given a normalized set of amplitudes $(\{a_i\})$ associated with basins $(\{R_i\})$, the formal framework states a **Born-style weighting rule**:

- The probability (or long-run frequency) with which outcome (R_i) is realized is proportional to $(|a_i|^2)$.

Crucially, in the formal core:

- This is **not** introduced as an extra, independent postulate.
- Instead, it arises as a **structural fact** from:
 - The way IN basins partition the attractor,
 - The way amplitudes on (\mathcal{P}) relate to those basins, and
 - Consistency requirements on how probabilities must compose across contexts and frames if they are to track the measure of IN basins.

In other words, once:

- Present amplitudes (v_i) on (\mathcal{P}) are tied to basins (R_i) on (IN), and
- A few structural conditions are imposed (e.g., context invariance, additivity under coarse-graining),

the squared norms ($|a_i|^2$) become the unique natural candidates for outcome weights. This is what is meant by **structural Born rule** in V1: the Born-style weighting is derived from present-structure, not appended by hand.

Collapse as structural selection at the hinge

In this formal picture:

- A **superposition** at the hinge corresponds to a situation where multiple basins ($\{R_i\}$) are co-eligible and carry nonzero amplitudes ($\{v_i\}$).
- A **collapse event** is described as:
 - Selecting one basin (R_j) according to the structural Born weights (proportional to $|a_j|^2$),
 - Resetting the amplitude assignment so that the state is now supported on the chosen basin (aligning the present-plane vector with (R_j)),
 - Updating (IN) so that (R_j) becomes part of the committed record.

This “collapse” is not postulated as a separate physical mechanism; it is a description of how, at the hinge, one outcome is selected and promoted from potential to record, consistent with the structural Born rule.

Compatibility with flip algebra and context ladder

The present-plane structure ((\mathcal{P}, J)) is not an independent add-on:

- It must be compatible with the **flip algebra** (so that the way amplitudes evolve along operator words matches the combinatorics of record/potential changes).
- It must be compatible with the **context ladder and pivot function** (so that gate weights derived from ($g(D)$) act linearly on amplitudes and commute with (J)).

This ensures that:

- The Born-style weighting respects the same hinge and pivot structure that governs the rest of the theory.
- Gate effects (including context-dependent attenuation or enhancement away from the hinge) can be expressed as scalar factors acting on amplitudes in (\mathcal{P}) , leaving the basic amplitude and Born machinery intact.

Together, the Present Plane and structural Born rule provide the minimal quantum-like component of the formal core: a way to encode interference and probabilistic selection at the hinge using amplitudes and squared norms, fully integrated into the present-act, context-ladder, and feasibility geometry structure, and without elevating Hilbert spaces and axiomatic Born rules to primitive status.

3.7 Context Ladder ($D(n)$) and Memory Dimension

The formal core associates the qualitative context levels $(\dots -2, -1, 0, +1, +2, +3 \dots)$ with a **discrete ladder index** ($n \in \mathbb{Z}$) and a corresponding **dimension profile** ($D(n)$). This profile summarizes how “thick” or “thin” the Inner Network is at each context level, and it is complemented by a **memory dimension** ($D_{\text{mem}}(n)$) that describes how much of that inner structure is available as active record for the vantage.

Context ladder index (n)

The context ladder is represented formally as an indexed family of layers:

$$\left[\begin{array}{l} \{ \mathcal{L}_n \mid n \in \mathbb{Z} \}, \\ \end{array} \right]$$

where each (\mathcal{L}_n) is a context level in the abstract ladder. In practice, the physically relevant band used in this framework is a 6-level segment:

- $(n = -2, -1, 0, +1, +2, +3),$

corresponding to the inner machinery, hinge/organism scale, and outer containers described in the ontological context-level section. However, the formalism itself allows for more levels in either direction if needed for other applications.

Dimension profile ($D(n)$)

To each context level (\mathcal{L}_n) , the framework assigns an effective inner dimension ($D(n)$), defined in the same sense as the inner dimension (D_k) associated with a particular Inner Network:

- $(D(n))$ is a scalar in $([D_{\min}, D_{\max}])$ that summarizes how the Inner Network associated with level (n) fills out its available configuration space.
- Informally, $(D(n))$ tells us whether the inner structure at that level is:

- Sparse/line-like ($(D(n))$ closer to 1),
- Sheet-like ($(D(n))$ near the hinge value $(D_{\text{ast}} \approx 2)$), or
- Volume-filling ($(D(n))$ closer to 3).

The dimension profile is not arbitrary; it must be consistent with:

- The relational roles of the levels (inner machinery vs hinge vs containers).
- The way streams of present-acts are nested across levels.
- The pivot function $(g(D))$, which singles out hinge-like behaviour near $(D \approx D_{\text{ast}})$.

In particular, the framework requires that there exists at least one context level (n) (designated as 0 in this work) such that:

$$\left[\begin{array}{l} D(0) \approx D_{\text{ast}} \approx 2, \\ \end{array} \right]$$

making level 0 the **hinge level** where inner geometry is ideally balanced between sparsity and overfilling.

Memory dimension $(D_{\text{mem}}(n))$

In addition to the total inner dimension $(D(n))$, the framework introduces a **memory dimension** $(D_{\text{mem}}(n))$ to describe how much of the inner structure at level (n) is accessible as **active record** for the vantage:

- $(D_{\text{mem}}(n))$ is defined such that

$$\left[\begin{array}{l} 0 \leq D_{\text{mem}}(n) \leq D(n), \\ \end{array} \right]$$
 for each (n) .
- When $(D_{\text{mem}}(n))$ is close to $(D(n))$, most of the inner structure at that level is “available” as record for present-acts at the Center.
- When $(D_{\text{mem}}(n))$ is significantly less than $(D(n))$, much of the inner structure at that level is effectively hidden or compressed from the vantage’s perspective.

Intuitively:

- At deep inner levels (e.g., $(n = -2, -1)$), we expect relatively high $(D(n))$ (rich substrate) but comparatively lower $(D_{\text{mem}}(n))$, because only compressed summaries of that machinery are accessible to Level 0 as record.
- At the hinge level $((n = 0))$, we expect:

$$\left[\begin{array}{l} D_{\{\text{mem}\}}(0) \approx D(0), \\ \end{array} \right]$$

reflecting that Level 0 is where active experience and record are maximally aligned. This is the level at which the present-act is centered.

- At outer container levels (e.g., $(n = +1, +2, +3)$), the inner structure may again be rich (e.g., environment, galaxy, cosmic structure), but $(D_{\{\text{mem}\}}(n))$ is limited by how much of that structure can be encoded into the vantage's present-acts as accessible record at any given tick.

Hinge conditions in terms of $(D(n))$ and $(D_{\{\text{mem}\}}(n))$

The hinge role of level 0 can be characterized formally by conditions such as:

- **Balance of inner and memory dimension:**

$$\left[\begin{array}{l} D(0) \approx D_{\text{ast}}, \quad D_{\{\text{mem}\}}(0) \approx D(0), \\ \end{array} \right]$$

indicating that Level 0 has inner geometry near the hinge dimension and that most of this geometry is available as record.

- **Asymmetry of inner vs outer:**
 - For inner levels (negative (n)), $(D(n))$ may approach high values (rich micro-structure), but $(D_{\{\text{mem}\}}(n))$ is significantly smaller, reflecting compression into effective variables at the hinge.
 - For outer levels (positive (n)), $(D(n))$ represents the structure of containers, while $(D_{\{\text{mem}\}}(n))$ reflects only the coarse-grained aspects of those containers that reach the Center.

These conditions ensure that:

- Level 0 is the unique (or at least distinguished) level where present-acts have their maximal access to inner structure as record.
- Inner levels behave as **machinery** that supplies dynamics but is not fully accessible to experience.
- Outer levels behave as **containers** whose detailed inner structure is largely inaccessible, with only coarse features recorded at any given tick.

Role in later constructions

The context ladder $(D(n))$ and memory dimension $(D_{\{\text{mem}\}}(n))$ support several later structures:

- In the **Context Level Framework**, they are used to map abstract levels (n) to physical scale bands (nano, micron, UGM, Earth, galactic, cosmic), guided by where (D(n)) and (D_{\text{mem}}(n)) exhibit hinge-like patterns and transitions.
- In the **gravity-as-feasibility** construction, the way (D(n)) varies across levels influences how feasibility gradients (implemented later via ParentGate in V2) are anchored at particular scales.
- In the **engine implementation**, manifest parameters such as feature resolutions and gate thresholds are tuned so that effective inner and memory dimensions at different bands respect the qualitative structure dictated by (D(n)) and (D_{\text{mem}}(n)).

Any implementation of the framework that claims fidelity to the formal core (V1) is expected to realize, at least in an effective sense, a dimension profile (D(n)) and memory-dimension profile (D_{\text{mem}}(n)) that satisfy these hinge conditions and support the role distinctions between inner machinery, hinge, and containers.

3.8 Terminology Note: “Division-by-Zero Operator”

In some of the longer explanatory documents, a particular formal operator is informally referred to as the “**division-by-zero operator**.” For the purposes of this defensive publication, and to avoid confusion, this operator will be referred to as the **Hinge Projection Operator**, with “division-by-zero” treated strictly as a historical nickname.

Formal role of the Hinge Projection Operator

The Hinge Projection Operator is a structural operator in the V1 formal framework that:

- **Localizes** the context-time action to a thin hinge region of the Present-Moment Sphere, concentrating dynamics onto the effective 2D boundary where IN and ON meet.
- **Promotes** this localized action to a 4D field-like description by “thickening” the hinge into a neighbourhood in which field-like effects (such as feasibility gradients or effective curvature) can be represented.

Informally, one can think of it as:

- Taking a context-time action that would otherwise be spread over a broader inner structure,
- Projecting it onto the hinge (a 2D-effective surface), and then
- Re-expressing it as a localized field action in the surrounding region.

This is a purely structural operation in the formal algebra; it does not perform arithmetic division of any quantity by zero.

Why “division-by-zero” appears as a nickname

The nickname “division-by-zero” arose from a philosophical interpretation in which:

- The hinge projection is seen as a formal analogue of mapping an “infinite context” (associated with the Infinite Present) onto a finite slice.
- The notation “1/0” is used metaphorically to signal a transition from a context in which any finite normalization fails (infinite context) to a finite, normalized local description.

In the formal, technical sense, however:

- No literal operation of dividing a number by zero is ever performed.
- The operator is defined as a mapping in the carrier and interval algebra, not as a numerical division.

For clarity and to avoid misinterpretation in technical and legal contexts, this defensive publication uses the neutral term **Hinge Projection Operator** and treats “division-by-zero” as an informal label only.

Relationship to other structures

The Hinge Projection Operator interacts with:

- The **invariant interval** (Section 3.4), by determining where and how the context-time action is concentrated on the hinge before being expressed in interval form.
- The **fractal inner geometry and pivot function** ($g(D)$) (Section 3.5), by acting most naturally when the inner dimension is near the hinge value (effective dimension close to 2).
- The **context ladder** (Section 3.7), by marking which levels behave as hinge-like surfaces versus bulk-like interiors or exteriors.

Any implementation of the formal framework (V1) that references a “division-by-zero operator” is, in this defensive publication, to be understood as implementing this **Hinge Projection Operator**—a structural localization/thickening mapping within the operator algebra, and not an arithmetic division-by-zero.

4. Present-Act Engine (V2) – Algorithmic Specification

4.1 Engine State at a Site (k)

The V2 present-act engine realizes the formal framework (V1) as a stepwise update over **discrete sites** indexed by an integer (k). At each site, the engine holds a complete, finite description of the current present-act for the vantage, together with the information needed to compute the next step.

Site index

- The engine evolves in discrete steps ($k \in \mathbb{Z}$).

- A single step (k to $k+1$) corresponds to one admissible Present-Act update for the vantage.
- There are no intermediate times between (k) and ($k+1$) at the control level.

Core state components at site (k)

At each site (k), the engine state consists of:

1. World record (W_k)

- A finite set (or multiset) of **world candidates** representing the outer configuration for the present-act.
- Each element ($w \in W_k$) carries:
 - A reference to an underlying configuration (e.g., a discretized environment/body state).
 - Discrete tags used for feature extraction (e.g., band, shell index, role flags).
- (W_k) is the engine's concrete representation of (ON_k) and the outer portion of the PMS at tick (k), compressed to the resolution allowed by the manifest.

2. Qualia record (Q_k)

- A finite set (or multiset) of **qualia candidates** representing the inner configuration for the present-act.
- Each element ($q \in Q_k$) carries:
 - A representation of a qualia-pixel (e.g., a local piece of sensory, proprioceptive, or internal state).
 - Discrete tags matching the feature alphabet (\mathcal{X}_i) (modality, location bins, intensity bins, phase bins, context tags).
- (Q_k) is the engine's concrete representation of (IN_k) as it is accessible to the present-act; deeper structure in levels -1 and -2 is compressed into these pixels.

3. Typed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$)

- A triple of typed budget variables, corresponding to:
 - ($\Delta \tau_k$): proper-time-like budget, derived from record-advancing flips.
 - (Δt_k): coordinate-time-like budget, derived from tick count.
 - (Δx_k): space-like budget, derived from configuration-change flips.
- These budgets are constrained at the control level to satisfy the discrete invariant-interval relation

$$\left[\begin{aligned} \Delta t_k^2 &= \Delta \tau_k^2 + \frac{\Delta x_k^2}{c^2}, \end{aligned} \right]$$
 for a fixed characteristic speed (c).
- The budgets are part of the engine state, not merely diagnostics; later gates may directly test whether candidate transitions respect budget constraints.

4. Context-level tags and band indices

- Each element of (W_k) and (Q_k) is tagged with context-level information (e.g., which band $-2, -1, 0, +1, +2, +3$ it belongs to, and what role it plays: inner machinery, hinge, container).
 - The site as a whole carries:
 - A current Center band (typically 0), indicating the vantage level.
 - References to active inner bands (e.g., $-1, -2$) and outer bands ($+1, +2, +3$) relevant for this step.
 - These tags are used by gates and by the ParentGate gravity mechanism to apply band-specific feasibility rules.
5. **Frame and manifest references**
- Each site (k) holds pointers to:
 - The current **frame** (collective synchronization structure) within which budgets and intervals are interpreted.
 - The **manifest**, a fixed configuration object that declares:
 - The feature alphabet (\mathcal{X}_i) and its internal structure.
 - The definitions of any feature maps (f_k, g_k).
 - The context-level band boundaries and hinge scales.
 - Gate thresholds, schedules, and any other engine parameters that must be fixed for the engine to be well-defined.
 - The manifest is considered part of the theory; it is not an arbitrary runtime configuration file.
6. **Engine-internal bookkeeping**
- Additional internal variables needed by the engine (e.g., random-seed state for PF/Born ties-only, counters for diagnostics, or cached adjacency structures) may be stored at the site level, but:
 - They must respect finiteness and locality constraints.
 - They must not introduce continuous control parameters into the fundamental update path.

State integrity constraints

At every site (k), the engine state must satisfy a set of **integrity conditions** before any update is applied:

- **Finiteness:**
(W_k) and (Q_k) are finite; their sizes are bounded above by manifest-specified limits.
- **Locality:**
Candidates in (W_k) and (Q_k) describe configurations that are local with respect to the current frame and context-level tags; no candidate encodes a nonlocal “jump” that would violate the no-skip rule.
- **Typed-budget consistency:**
The triple ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$) satisfies the invariant-interval relation within tolerances specified by the manifest.
- **Context-tag consistency:**
Context-level tags are coherent across (W_k), (Q_k), and any shared frame/CS structures: inner, hinge, and container roles are assigned in a way compatible with the CL mapping.

Only when these integrity conditions hold is the site $(\mathcal{S})_k = (W_k, Q_k, \Delta \tau_k, \Delta t_k, \Delta x_k, \text{tags}, \text{manifest ref}, \text{frame ref})$ considered a valid state on which the engine cycle (enumeration, hinge equality, feasibility gates, ratio-lex, PF/Born ties-only, commit) may be applied to produce the next site $(\mathcal{S})_{k+1}$.

4.2 Feature Alphabet Ξ and Hinge Equality

The present-act engine uses the **Finite Feature Alphabet** (Ξ) to encode both inner (qualia) and outer (world) content in a common discrete vocabulary. At the engine level, (Ξ) and its associated feature maps are part of the manifest and are treated as fixed structural data for a given configuration.

Feature alphabet Ξ at the engine level

- (Ξ) is a finite set of feature tokens. Each token encodes a small, structured unit of content (inner or outer) in terms of discrete tags such as:
 - modality (e.g., visual, auditory, proprioceptive, interoceptive),
 - spatial bin (e.g., angle and distance bins around the Center, or body/environmental coordinates),
 - intensity/magnitude bin,
 - phase bin (discrete phase of interference-like structure),
 - context/band tags (which context level $-2 \dots +3$ and what role),
 - and any further discrete attributes required by the manifest.
- The manifest specifies:
 - the finite set of allowed tokens in (Ξ) ,
 - the internal structure of each token (which tags it carries),
 - and any constraints on which combinations of tags are legal.

The control path of the engine is required to depend **only** on discrete predicates over tokens in (Ξ) and their relations. Continuous-valued fields or metrics may exist in diagnostics, but they are not used in the core decision-making steps.

Feature maps from world and qualia records

At each site (k) , the engine uses two manifest-defined feature maps:

- An outer feature map

$$[f_k : W_k \rightarrow \Xi,$$

$$]$$
which assigns to each world candidate $(w \in W_k)$ a feature token $(f_k(w) \in \Xi)$.
- An inner feature map

$$[g_k : Q_k \rightarrow \Xi,$$

$$]$$
which assigns to each qualia candidate $(q \in Q_k)$ a feature token $(g_k(q) \in \Xi)$.

These maps are:

- **Fixed by the manifest:** they do not adapt or learn during engine operation.
- **Local and band-aware:** the construction of $(f_k(w))$ and $(g_k(q))$ may depend on local context (e.g., which band a candidate belongs to, local frame), but not on global, nonlocal information.
- **Finite:** for any given (k) , (W_k) and (Q_k) are finite, and so are their images under (f_k) and (g_k) .

The same alphabet $(\backslash X_i)$ and the same tagging conventions are used for both inner and outer content, so that the engine can compare them directly at the hinge.

Hinge equality as the PMS boundary rule

The PMS boundary condition in the engine is implemented as a **hinge equality** test in $(\backslash X_i)$. For a given site (k) , candidate pairs $((w_{k+1}, q_k))$ are constructed (later in the pipeline) and tested for boundary compatibility via:

$$\left[\begin{array}{l} f_{k+1}(w_{k+1}) = g_k(q_k). \end{array} \right]$$

A pair $((w_{k+1}, q_k))$ is **hinge-admissible** if and only if:

- The outer feature token for the candidate next world (w_{k+1}) exactly matches
- The inner feature token for the current qualia candidate (q_k) ,

under the maps declared in the manifest.

This implements the PMS boundary condition as a strict equality in the feature alphabet:

- There is **no** use of metric similarity, distance thresholds, cosine similarity, or any other continuous scoring function in the fundamental hinge condition.
- Either the feature tokens match exactly (admissible at the hinge), or they do not (rejected at the hinge).

The strictness of this condition is central to the framework:

- It ensures that any committed outer configuration is **qualia-compatible** with the inner state at the moment of commitment.
- It removes any ambiguity about “partial matches” or “almost the same” at the fundamental level; such notions can only enter, if at all, through higher-level coding choices in $(\backslash X_i)$, not through the hinge rule itself.

Multi-pixel and composite hinge conditions

In practice, present-acts often involve structured patterns (e.g., multiple qualia-pixels and corresponding world-pixels). The framework allows for **composite hinge conditions** defined over small finite sets:

- A pattern of inner candidates ($\{q_k^{(i)}\}$) and outer candidates ($\{w_{k+1}^{(j)}\}$) can be considered as a unit, provided the manifest specifies how to form composite tokens or composite match conditions.
- In that case, hinge equality is still expressed in terms of equality of feature tokens in $(\backslash X_i)$ or equality of structured combinations of such tokens, not in terms of metric closeness.

For the purposes of this defensive publication, it is enough to state that:

- Every admissible candidate transition at the PMS boundary must satisfy a hinge condition expressible as equality over $(\backslash X_i)$, possibly extended to finite tuples.
- There is no admissible fundamental transition that bypasses this equality test.

Interaction with typed budgets and gates

The hinge equality filter is applied **before** feasibility gates, budget checks, and stochastic resolution:

1. Candidate pairs that fail the hinge equality test are discarded and play no further role in the control path.
2. Only hinge-admissible pairs are passed forward to:
 - time, structural, and context gates,
 - ParentGate (gravity-related feasibility),
 - ratio-lexicographic acceptance,
 - PF/Born ties-only randomness.

Because all later steps operate only on candidates that are already hinge-compatible, all engine decisions are grounded in relationships between inner and outer content encoded in $(\backslash X_i)$. This is one of the main reasons the framework qualifies as **qualia-first** at the algorithmic level: what can happen next is constrained from the outset by exact matches between inner qualia-encodings and outer world-encodings, rather than by abstract material configurations alone.

4.3 Engine Cycle: High-Level Pipeline

At each site (k) , the present-act engine applies a fixed, finite pipeline to advance the state from (\mathcal{S}_k) to (\mathcal{S}_{k+1}) . This pipeline is the concrete realization of the V1 tick algebra (Renew, Sink, Trade, Sync, Framing) in terms of discrete operations on $((W_k, Q_k))$, the feature alphabet $(\backslash X_i)$, feasibility gates, and typed budgets.

For clarity, the core cycle can be described as the following steps:

1. **Selector phase: build the candidate set**
2. **Hinge equality: enforce PMS boundary compatibility**

3. **Feasibility gating:** Θ , κ , structural, ParentGate, CRA-like gates
4. **Ratio-lexicographic comparison:** discrete residual ordering
5. **Fewest-acts tiebreak:** discrete geodesic preference
6. **PF/Born ties-only:** stochastic choice on exact ties
7. **Commit and budget update:** form (\mathcal{S}_{k+1})
8. **Diagnostics and audits** (outside control, but attached to the same cycle)

Each step uses only finite sets, discrete tags, and integer/ratio operations in control; continuous quantities (if any) are confined to diagnostics and are not allowed to influence the core decision logic.

Step 1 – Selector phase: build the candidate set

Given the state (\mathcal{S}_k) :

- The engine uses **selectors** to build a finite set of candidate world–qualia pairs:

$$[\mathcal{C}_k = \{c_i\} = \{(w_{k+1}^{(i)}, q_k^{(i)})\}.]$$
- Selectors are purely relational rules (e.g., local adjacency, context compatibility, band-wise contiguity), specified in the manifest. They:
 - Enumerate possible local moves in the world record (W_k) that are allowed by locality and no-skip constraints.
 - Pair those moves with corresponding qualia candidates in (Q_k) that represent the inner side of the same potential act (e.g., “seeing this”, “feeling that”).
- No probabilities, continuous scores, or learned weights are used in the selector phase. The output is simply a finite set (\mathcal{C}_k) of relationally generated candidates.

If (\mathcal{C}_k) is empty, the act is blocked at this site (no admissible continuation); handling of such corner cases is specified in the engine configuration but does not alter the structure of the pipeline.

Step 2 – Hinge equality: enforce PMS boundary compatibility

For each candidate $(c = (w_{k+1}, q_k) \in \mathcal{C}_k)$, the engine:

- Computes feature tokens:

$$[\xi_{\text{out}} = f_{k+1}(w_{k+1}), \quad \xi_{\text{in}} = g_k(q_k),]$$
 using the manifest-declared maps into (\mathcal{X}_i) .

- Applies the **hinge equality** rule:

$$\begin{aligned} &[\\ &f_{k+1}(w_{k+1}) = g_k(q_k) \\ &] \end{aligned}$$

(or the equivalent composite condition, if the manifest uses multi-pixel tokens).

Any candidate that fails hinge equality is discarded at this stage. Only candidates that satisfy exact equality in (Ξ) are passed forward. There is no metric similarity or tolerance threshold in this test.

Let the survivors of this step be:

$$\begin{aligned} &[\\ &\mathcal{C}_k^{\{\text{hinge}\}} \subseteq \mathcal{C}_k. \\ &] \end{aligned}$$

If $(\mathcal{C}_k^{\{\text{hinge}\}} = \varnothing)$, the act is blocked; no further steps are applied.

Step 3 – Feasibility gating: Θ , κ , structural, ParentGate, CRA-like gates

On the hinge-admissible set $(\mathcal{C}_k^{\{\text{hinge}\}})$, the engine applies a fixed sequence of **feasibility gates**. Each gate is a boolean or ordinal function of a candidate's discrete tags and local structure; no continuous weights are used.

The standard gate sequence is:

1. **Θ (time-stability gate)**
 - Checks whether the candidate is compatible with temporal stability constraints at the relevant bands (e.g., respecting the temporal hinge, not forcing unrealistically rapid changes).
 - Operates on band tags, budgets, and local temporal structure.
2. **κ (granularity/persistence gate)**
 - Ensures that changes in discrete degrees of freedom (e.g., positions, feature resolutions) occur at rates consistent with granularity and persistence conditions.
 - Prevents pathological “flickering” or abrupt resolution shifts that violate the manifest's granularity rules.
3. **Structural gates**
 - Enforce graph-based and pattern-based conditions:
 - Contiguity and connectivity in local neighbourhoods,
 - Degree constraints,
 - Orientation and L3 (container) persistence conditions,
 - Symmetry/consistency with the current frame and CS.
 - All outputs are boolean or ordinal (e.g., “passes/ fails”, “degree within bounds / out of bounds”).
4. **ParentGate (gravity schedule)**

- Applies radius- and band-dependent feasibility constraints tied to large-scale containers (e.g., planet, star, galaxy).
- Uses only:
 - Shell index,
 - Integer strictness level per shell,
 - Local combinatorial patterns (e.g., how mass/structure moves between shells).
- Returns a boolean pass/fail; it is the **only** gate allowed to depend directly on large-scale radial position.

5. CRA-like gates (Context-Resolved Admissibility)

- Ensure that candidates respect context partitions (e.g., distinctions between different lanes, context IDs, or band families).
- Prevent “common-mode” commits where two structurally distinct context families would be collapsed without justification.
- Operate on discrete context tags and band/lane identifiers.

All of these gates are:

- Declared in the manifest,
- Boolean or finite-ordinal,
- Applied in a fixed order, and
- Operate only on discrete tags and local structure.

Let the survivors after all gates be:

[
 $\mathcal{C}_k^{\text{feas}} \subseteq \mathcal{C}_k^{\text{hinge}}$.
]

If $(\mathcal{C}_k^{\text{feas}} = \text{varnothing})$, the act is blocked or special fallback rules are applied as specified in the manifest; no candidate proceeds to acceptance.

Step 4 – Ratio-lexicographic comparison: discrete residual ordering

For each candidate $(c \in \mathcal{C}_k^{\text{feas}})$, the engine computes a **residual triple**:

[
 $\mathbf{d}(c) = (\mathbf{d}_{\text{out}}(c), \mathbf{d}_{\text{in}}(c), \mathbf{d}_{\text{times}}(c))$,
]

where each component is a ratio of failed checks to total checks in a given channel, such as:

- $(\mathbf{d}_{\text{out}}(c))$: fraction of outward (world-side) structural checks that failed,
- $(\mathbf{d}_{\text{in}}(c))$: fraction of inward (qualia-side) checks that failed,

- $(d_{\text{times}}(c))$: fraction of cross checks (hinge consistency, context alignment, etc.) that failed.

Each ratio is computed as:

$$\left[\frac{\text{failed_checks}}{\text{total_checks}}, \right]$$

and lies in $[0, 1]$. These ratios are derived purely from discrete counts; no learned weights or continuous scoring functions are introduced.

The engine then applies a **lexicographic ordering** on these residual triples:

- Candidates with smaller (d_{out}) are preferred; if tied, smaller (d_{in}) is preferred; if still tied, smaller (d_{times}) is preferred.
- This induces a strict ordering or a set of ties on $(\mathcal{C}_k^{\text{feas}})$.

This step is entirely deterministic and discrete.

Step 5 – Fewest-acts tiebreak: discrete geodesic preference

If, after lexicographic comparison, there remains a non-trivial tie set $(\mathcal{T} \subseteq \mathcal{C}_k^{\text{feas}})$ (i.e., two or more candidates with identical residual triples), the engine applies a **fewest-acts tiebreak**:

- For each $(c \in \mathcal{T})$, the engine computes an integer “act count” or similar discrete path-length measure (how many ticks, or how many structural changes, this candidate implies relative to a baseline).
- It selects the candidate(s) with minimal act count as preferred; this corresponds to a discrete “geodesic” preference for simpler/shorter feasible continuations.

If this step singles out a unique candidate, that candidate is chosen as the survivor; no randomness is used. If multiple candidates remain tied (same residual triple and same act count), the engine proceeds to the PF/Born ties-only step.

The fewest-acts measure:

- Is integer-valued,
- Is specified up front in the manifest,
- Does not depend on any continuous weights or fitted parameters.

Step 6 – PF/Born ties-only: stochastic choice on exact ties

Only when **all** discrete ordering steps fail to produce a unique winner:

- Residual triples are identical (within specified ε -zeroing),
- Fewest-acts tiebreak has not singled out a unique candidate,
- The tie set (\mathcal{T}) has size ($|\mathcal{T}| \geq 2$),

does the engine invoke the **PF/Born ties-only** mechanism.

In that case:

1. The engine builds a small adjacency graph on (\mathcal{T}), representing which candidates are considered “coherent” with one another (e.g., based on shared phase-bin structure in (Ξ) and local structural relations).
2. It constructs a primitive column-stochastic kernel (M) on (\mathcal{T}), with entries determined purely by this adjacency and simple combinatorial rules declared in the manifest.
3. It computes the Perron–Frobenius eigenvector (v) of (M).
4. It defines Born-style weights ($w_j \propto v_j^2$) for each candidate ($j \in \mathcal{T}$), normalized so ($\sum_j w_j = 1$).
5. It uses a specified RNG (with declared seed family) to sample one candidate from the distribution ($\{w_j\}$).

Key points:

- PF/Born is **only** used on exact ties after all discrete ordering steps.
- The kernel (M) and weights ($\{w_j\}$) are determined entirely by local adjacency and manifest-declared rules, not by fitted parameters.
- This step is the sole source of randomness in the control path and is the engine-level realization of the structural Born rule from V1.

Step 7 – Commit and budget update: form (\mathcal{S}_{k+1})

Once a single winner ($c^{\ast} = (w_{k+1}^{\ast}, q_k^{\ast})$) has been selected (deterministically or via PF/Born), the engine:

- Commits (w_{k+1}^{\ast}) as the next world configuration and updates (W_{k+1}) accordingly.
- Updates (Q_{k+1}) to reflect the new inner record, incorporating whatever qualia changes are implied by the act.
- Computes the typed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$) for the transition ($k \rightarrow k+1$) from flip counts or gate outcomes, enforcing the invariant-interval relation:
[
 $\Delta t_k^2 = \Delta \tau_k^2 + \frac{\Delta x_k^2}{c^2}$.
]

- Advances the site index to $(k+1)$ and assembles the new state (\mathcal{S}_{k+1}) with updated records, budgets, and context tags.

This completes a single engine cycle.

Step 8 – Diagnostics and audits

In parallel with or immediately after the commit step, the engine (or associated tooling) may:

- Record diagnostic data about gate outcomes, ParentGate acceptance rates, and PF/Born tie events.
- Run audits to verify:
 - Finiteness and locality (no-skip rule),
 - Curve-ban compliance (no continuous weights in control),
 - Invariant-interval consistency of budgets,
 - ParentGate symmetry properties (e.g., rotation invariance in expectation),
 - Context-resolved admissibility (no unintended collapsing of distinct contexts).

These diagnostics and audits are not part of the selection logic itself—they do not feed back into the control path—but they are part of the declared implementation contract and must be available to demonstrate that an engine instance conforms to the framework’s structural constraints.

Together, these eight steps define the **engine cycle** for the present-act engine. Any implementation that claims to instantiate the V2 framework must, at minimum, realize an update rule equivalent to this pipeline, using finite sets, discrete tags in (\mathcal{X}_i) , fixed gate sequences, ratio-lexicographic ordering, a discrete fewest-acts tiebreak, and PF/Born randomness restricted to exact ties.

4.4 Feasibility Gates (Non-Gravity)

This subsection specifies the **non-gravity feasibility gates** used in the engine cycle before any gravity-related strictness (ParentGate) is applied. These gates are responsible for enforcing local temporal, granular, and structural consistency, independent of large-scale gravitational effects.

All gates in this subsection:

- Operate only on **finite sets**, **discrete tags**, and **integer/ratio counts**.
- Are declared in the **manifest** (no runtime learning).
- Return **boolean** or **finite-ordinal** outcomes.
- Do **not** encode gravity; gravity-specific feasibility is handled separately via ParentGate (described later in the gravity section).

For any hinge-admissible candidate $(c = (w_{k+1}, q_k))$, the engine applies the following gates in a fixed order.

4.4.1 Θ Gate – Temporal Stability and Hinge Window

The Θ gate enforces compatibility with the **temporal hinge** and band-wise stability constraints.

Inputs:

- Typed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$) for the transition ($k \rightarrow k+1$).
- Context-level tags (e.g., whether the change is primarily at $-2, -1, 0, +1$, etc.).
- Manifest parameters for temporal windows at each band (including the temporal hinge scale at Level 0).

Checks (examples):

- The implied (Δt) for the act is within the admissible window for the relevant band(s).
- Inner-band changes (e.g., at $-2, -1$) do not demand time steps shorter than the minimum temporal resolution declared in the manifest.
- For Level 0 (hinge), the act fits within the allowed integration window around the temporal hinge (e.g., on the order of ~ 0.1 s in physical units once mapped).

Output:

- **Pass/Fail.**
- Optionally, a small finite ordinal (e.g., “good fit”, “borderline”, “poor fit”) that may contribute to later residual ratios.

No gravity or large-scale geometric data enter into Θ ; it purely enforces **temporal coherence** relative to band-wise constraints and the invariant interval.

4.4.2 κ Gate – Granularity and Persistence

The κ gate enforces **granularity** (discrete step sizes) and **persistence** (no pathological flicker) in both inner and outer configurations.

Inputs:

- Discrete configuration differences between (W_k) and (W_{k+1}).
- Discrete configuration differences between (Q_k) and the implied (Q_{k+1}) for the candidate.
- Manifest parameters specifying minimal and maximal allowed discrete changes for each feature tag (e.g., position bins, intensity bins, phase bins).

Checks (examples):

- Spatial bin changes do not exceed band-specific step limits (e.g., one or a few bins per tick).
- Intensity changes remain within allowed discrete increments (no instantaneous jump from minimum to maximum).
- Phase-bin changes do not violate continuity constraints set for interference-like structure.
- Qualia-pixels and world-pixels do not “blink” on and off in a way that violates persistence rules (e.g., an object cannot disappear and reappear without stepping through intermediate states, unless explicitly allowed by a structural rule).

Output:

- **Pass/Fail**, and optionally a finite ordinal reflecting how close the candidate is to the ideal granularity/persistence behaviour.

κ does not depend on radial position, shells, or gravitational amplitude; it enforces **local discrete smoothness** of change.

4.4.3 Structural Gates – Local Graph and Pattern Constraints

The **structural gates** enforce constraints on the local relational structure encoded in (W_k) and (Q_k).

Inputs:

- Graph or hypergraph representation of local neighbourhoods in the world and qualia records.
- Band tags, frame information, and any object/role tags declared in the manifest.

Checks (examples):

- **Connectivity:**
 - Nodes representing parts of the same object remain connected under allowed patterns; the candidate does not arbitrarily break or join objects in ways prohibited by the manifest.
- **Degree constraints:**
 - Node degrees (e.g., how many neighbours each has) remain within allowed ranges; this prevents unnatural super-connectivity or fragmentation.
- **Orientation and alignment:**
 - For structured objects (e.g., oriented segments, layered patterns), changes in orientation fall within allowed discrete rotation/shift increments.
- **Frame compatibility:**
 - Candidate changes are consistent with the current frame’s synchronization: shared objects in a Collective Sphere are updated coherently across PMSs.

Output:

- **Pass/Fail** for each structural check category.
- Combined into an overall structural gate outcome (pass/fail and, if used, a small ordinal indicating quality).

These gates enforce **local structural integrity** and compatibility with shared frames and CSs, without referencing gravity or shell-based feasibility.

4.4.4 CRA-Like Gates – Context-Resolved Admissibility

The **Context-Resolved Admissibility (CRA-like)** gates enforce constraints on how distinct contexts are maintained and resolved.

Inputs:

- Context tags associated with candidates: context family IDs, lanes, or partition labels (e.g., different experimental setups, different “branches” of a higher-level context).
- Manifest-declared rules for when contexts may merge, split, or remain distinct.

Checks (examples):

- A candidate does not collapse two distinct context families into one outcome without an explicit rule permitting such a merge.
- Context IDs associated with qualia and world components remain aligned in ways consistent with the intended semantics (e.g., different “lanes” of a computation or different experimental contexts).
- Forbidden cross-context “short circuits” are not introduced (e.g., a world candidate from one context lane cannot be matched with a qualia candidate from an incompatible context lane).

Output:

- **Pass/Fail** for context-resolved admissibility.
- If used in residuals, an integer count of context-constraint violations.

These gates ensure that the engine does not unintentionally erase meaningful distinctions between contexts. They are particularly important in simulations where multiple scenarios or “lanes” are being explored in parallel.

4.4.5 Gate Composition and Non-Gravity Guarantee

The non-gravity feasibility gates are applied in a **fixed sequence** after hinge equality and before any gravity-specific gate:

1. Θ (temporal stability / hinge window)
2. κ (granularity / persistence)
3. Structural gates (graph/pattern constraints)
4. CRA-like gates (context-resolved admissibility)

Key guarantees:

- None of these gates depend on radial shells, gravitational amplitude χ , or large-scale context-ladder ratios.
- Their definitions in the manifest use only:
 - Band and frame tags,
 - Local structure in (W_k) and (Q_k) ,
 - Typed budgets and discrete flip counts,
 - Context IDs and lane/partition labels.

All gravity-related feasibility is reserved for **ParentGate**, which is specified separately in the gravity section and is the only gate permitted to depend directly on large-scale radial structure and χ .

Any engine implementation that claims to respect this framework must preserve this separation: temporal, granular, structural, and context-resolved feasibility must be enforced by non-gravity gates as described here, with gravity-specific effects confined to the dedicated ParentGate mechanism.

4.5 PF/Born Ties-Only Rule – Algorithmic Detail

The PF/Born ties-only procedure is the **only** source of randomness in the engine’s control path. It is invoked **only** when all purely discrete ordering steps (hinge equality, feasibility gates, ratio-lex ordering, and fewest-acts tiebreak) fail to select a unique winning candidate.

This subsection gives an enabling, algorithmic specification of the PF/Born ties-only rule.

4.5.1 Preconditions for invoking PF/Born

Let $(\mathcal{C}_k^{\text{feas}})$ be the set of candidates that have passed:

- hinge equality,
- all non-gravity feasibility gates,
- ParentGate (gravity gate, specified later), and
- basic structural/context checks.

Let the engine have computed for each candidate $(c \in \mathcal{C}_k^{\text{feas}})$:

- a residual triple $(\mathbf{d}(c) = (d_{\text{out}}, d_{\text{in}}, d_{\text{times}}))$, and

- a discrete “fewest-acts” measure ($A(c)$).

Define the lexicographic minimal residual value:

$$[\mathbf{d}\}_{\min} = \min_{\{\text{lex}\}} \{ \mathbf{d}(c) : c \in \mathcal{C}_{k^{\text{feas}}} \}.$$

Define the set of residual-minimal candidates:

$$[\mathcal{C}_{k^{\text{min}}}] = \{ c \in \mathcal{C}_{k^{\text{feas}}} : \mathbf{d}(c) = \mathbf{d}_{\min} \}.$$

Then:

- If $(|\mathcal{C}_{k^{\text{min}}}| = 0)$, the act is blocked (no admissible continuation).
- If $(|\mathcal{C}_{k^{\text{min}}}| = 1)$, that candidate is selected deterministically.
- If $(|\mathcal{C}_{k^{\text{min}}}| \geq 2)$, compute

$$A_{\min} = \min \{ A(c) : c \in \mathcal{C}_{k^{\text{min}}} \},$$
 and define

$$[\mathcal{T}] = \{ c \in \mathcal{C}_{k^{\text{min}}} : A(c) = A_{\min} \}.$$

The PF/Born ties-only rule is invoked **only if** $(|\mathcal{T}| \geq 2)$: a non-trivial tie remains after all discrete ordering steps.

4.5.2 Building the tie graph and kernel

Given a non-empty tie set $(\mathcal{T} = \{ c_1, \dots, c_N \})$ with $(N \geq 2)$, the engine constructs:

1. A **tie adjacency graph** $(G = (V, E))$, where:
 - $(V = \mathcal{T})$ (each node is a candidate).
 - Edges (E) are defined by a manifest-declared **coherence rule** that depends only on:
 - feature tokens in (\mathbf{X}_i) for each candidate,
 - local structural relations (e.g., adjacency, phase-bin compatibility, band alignment), and
 - context tags (not on any fitted weights or continuous scores).

For example, two candidates (c_i) and (c_j) might be connected if:

- they are identical except for a small phase-tag difference, or
 - they differ only by a localized, manifest-allowed discrete shift in a feature bin.
2. A **primitive column-stochastic kernel** ($M \in \mathbb{R}^{N \times N}$), with entries ($M_{ij} \geq 0$) and ($\sum_i M_{ij} = 1$) for each column (j). (M) is constructed using a manifest-specified rule such as:
- Start from the adjacency structure of (G).
 - For each node (j), distribute probability mass among its neighbours (i) (and possibly itself) in fixed rational proportions based only on discrete graph properties (e.g., degree, band tags, phase-bin matches).
 - Ensure **primitivity** (some power of (M) has strictly positive entries) by including a small, uniform “leak” if necessary, defined in a purely combinatorial way.

No continuous parameters, fitted weights, or external training data are allowed in the construction of (M); all rules are declared in the manifest in terms of adjacency and discrete tags.

4.5.3 Perron–Frobenius eigenvector and Born weights

Once (M) is defined, the engine computes its unique Perron–Frobenius eigenvector (v):

- ($v \in \mathbb{R}^N$), ($v_i > 0$) for all (i).
- ($M v = \lambda_{\text{PF}} v$), with (λ_{PF}) the spectral radius.
- (v) is normalized so that ($\sum_i v_i = 1$) or any other fixed convention; the choice does not affect the final probabilities.

The engine then defines **Born weights**:

$$[w_i \propto v_i^2, \quad w_i \geq 0,]$$

with normalization:

$$[p_i = \frac{w_i}{\sum_{j=1}^N w_j}, \quad \sum_{i=1}^N p_i = 1.]$$

Here:

- The squared entries (v_i^2) provide the analogue of ($|a_i|^2$) in the structural Born rule.
- The mapping from adjacency \rightarrow kernel (M) \rightarrow eigenvector (v) \rightarrow squared weights is entirely determined by discrete, manifest-declared rules.

This defines a probability distribution ($\{p_i\}$) over the tie set (\mathcal{T}).

4.5.4 Random sampling and commitment

The final step is to use a random number generator (RNG) to select a winner from (\mathcal{T}) according to $(\{p_i\})$.

Requirements:

- The RNG must be a standard, well-documented generator with an explicit seeding scheme.
- The seed for each site (k) must be determined from:
 - a manifest-declared base seed, and
 - local, discrete information (e.g., tick index, hash of (W_k, Q_k) , and context tags),

such that:

- The PF/Born step is reproducible given the same initial seed and state.
- The randomness is not influenced by any external or hidden continuous variables.

Algorithm:

1. Compute a cumulative distribution from $(\{p_i\})$:
[
 $P_i = \sum_{j=1}^i p_j, \quad P_N = 1.$
]
2. Draw a uniform random variate (u) in $([0,1))$ from the RNG.
3. Select the unique index (i^{\ast}) such that $(P_{i^{\ast}-1} \leq u < P_{i^{\ast}})$ (with $(P_0 := 0)$).
4. Declare $(c^{\ast} = c_{i^{\ast}})$ the winner.
5. Proceed to the commit and budget-update step, forming (\mathcal{S}_{k+1}) based on (c^{\ast}) .

4.5.5 Ties-only and separation from other randomness

The PF/Born rule is strictly a **ties-only** mechanism:

- It is never used when a unique winner already exists after discrete ordering (hinge, gates, residuals, fewest-acts).
- It is never used to adjust or bias candidate sets prior to the tie condition; all such operations are fully deterministic.

No other source of randomness is allowed in the control path. In particular:

- Selectors, gates, ratio-lex comparison, and fewest-acts tiebreak are all deterministic given the state and manifest.
- Randomness cannot appear in the construction of (M) , (v) , or $(\{p_i\})$; only in the final sampling step.

Any engine implementation claiming fidelity to this framework must:

- Implement a PF/Born ties-only rule with the structural pattern described here (adjacency \rightarrow primitive kernel \rightarrow PF eigenvector \rightarrow squared weights \rightarrow RNG sampling), and
- Restrict its use to the exact-tie condition defined above, leaving the rest of the control path deterministic.

4.6 Implementation Principles (Finiteness, Locality, Curve-Ban, Auditability)

The present-act engine is distinguished from generic simulations by a small set of **non-negotiable implementation axioms**. These axioms are part of the claim: any implementation that omits or violates them is not considered an instance of the framework described in this defensive publication.

The core principles are:

- **Finiteness**
- **Locality (No-Skip Rule)**
- **Curve-Ban (no continuous control in the engine core)**
- **Auditability (Manifest + Mandatory Audits)**

4.6.1 Finiteness

The engine is required to operate entirely on **finite** structures in its control path:

- For every site (k) :
 - The world record (W_k) and qualia record (Q_k) are finite sets (or multisets).
 - The candidate set (\mathcal{C}_k) constructed in the selector phase is finite.
 - All intermediate candidate subsets (hinge-admissible, gated, tie sets) are finite.
- The feature alphabet (\mathcal{X}_i) is finite. No unbounded or continuous feature spaces are allowed in the control logic.
- Neighbourhoods used by selectors and structural gates are finite: only a bounded number of neighbours per element may be examined at each step.
- Typed budgets $((\Delta \tau_k, \Delta t_k, \Delta x_k))$ and residuals used by ratio-lex comparison are derived from **finite** integer counts.

No unbounded data structures (e.g., infinitely growing graphs, unbounded arrays, or continuous domains) are permitted **in the engine's decision path**. Long-running simulations may

accumulate data in external logs, but the engine's step-by-step update logic always acts on finite, bounded structures.

4.6.2 Locality and the No-Skip Rule

The engine enforces a strict **Locality** principle via a **No-Skip Rule**:

- Tick updates are strictly **nearest-neighbour in tick index**:
 - The fundamental update is $(k \rightarrow k+1)$.
 - There is no single primitive that jumps directly from (k) to $(k+m)$ for $(m>1)$ while bypassing the intermediate sites.
- Spatially (or structurally), selectors and gates may only use **local neighbourhoods** as defined by:
 - adjacency in the world record,
 - adjacency in the qualia record,
 - band/frame-based neighbourhood definitions declared in the manifest.
- Forbidden:
 - Any update that directly changes widely separated parts of the configuration in a single step without passing through intermediate configurations.
 - Any gate that depends on non-local aggregates that cannot be computed from a bounded local neighbourhood in a finite number of discrete operations.

The No-Skip Rule is one of the main constraints that distinguish this engine from generic physics simulations or neural networks that freely manipulate global state across large distances or time steps.

4.6.3 Curve-Ban: No Continuous Control in the Core

The **Curve-Ban** prohibits the use of continuous curves, floating-point weight fields, or “soft” activations in the **control path** of the engine. Concretely:

- Gates, selectors, and acceptance rules must rely only on:
 - discrete predicates (true/false),
 - integer or rational ratios constructed from finite counts,
 - ordered comparisons of these discrete quantities.
- Forbidden in the control path:
 - floating-point weight vectors that are updated via gradient descent or any continuous optimisation,
 - continuous-valued “attention” scores or similarity metrics that directly drive selection,
 - smooth activation functions (e.g., sigmoids, tanh, ReLUs) when used as decision layers in the engine.

Continuous quantities may still appear in **diagnostics** (e.g., plotting a curvature profile, estimating a fractal dimension for analysis), but any such diagnostic values:

- Must not feed back into gates, selectors, or acceptance rules.
- Must not be used to adjust manifest parameters or control logic at runtime.

The Curve-Ban ensures that the engine's decisions are entirely **combinatorial and ordinal**, not gradient-based or continuous-control in the style of mainstream machine learning.

4.6.4 Auditability: Manifest and Mandatory Audits

The framework requires that any engine implementation be **auditable** against a fixed, immutable **Manifest** and a small set of **mandatory audits**.

Manifest

- The Manifest is a **static, immutable configuration object** that forms part of the theory itself. It declares:
 - the feature alphabet (\mathcal{X}) and its internal tag structure,
 - the definitions of the feature maps (f_k) and (g_k),
 - context-level band boundaries and hinge scales (e.g., UGM, temporal hinge),
 - gate definitions and thresholds,
 - ParentGate shell structure and gravity-related parameters (including χ),
 - diagnostics and audit parameters.
- The engine is **prohibited** from modifying manifest-declared values at runtime. To change the Manifest is to change the theory version, not to “learn” within the same configuration.

Mandatory audits

To distinguish this system from standard neural networks or generic simulations, the following audits are required:

1. Curve-Ban Audit

- A static analysis that checks:
 - No floating-point curves, learned weight fields, or soft activations appear in the control path.
 - All decisions (selector, gate, acceptance) are derived from discrete predicates and fixed combinatorial rules.

2. Diagnostics-Leak Ban

- A taint-style analysis that ensures:
 - No data originating in diagnostic or observation modules (e.g., continuous metric estimates, external measurements used only for logging) flows back into the engine's control logic.

- The control path depends solely on the current site state $((W_k, Q_k, \text{budgets}, \text{tags}))$ and the Manifest, not on arbitrary diagnostic channels.

3. PF/Born Integrity Audit

- A check that:
 - Probabilistic selection (RNG use) is **only** triggered under the exact-tie condition defined in the PF/Born ties-only rule.
 - No other parts of the control path introduce randomness.
 - The kernel (M), PF eigenvector (v), and squared weights (w_i) are constructed exactly as specified, without external or hidden tuning.

Additional audits may be added in specific implementations (e.g., locality checks, invariant-interval consistency), but these three classes are treated as **mandatory**.

4.6.5 Summary of Implementation Contract

Taken together, these principles define the **implementation contract** for any engine claiming to instantiate the V2 present-act framework:

- All control logic operates on **finite** sets and tags.
- Tick updates and spatial changes respect **locality** and the **No-Skip Rule**.
- The control path obeys a strict **Curve-Ban**, avoiding continuous control mechanisms.
- A fixed **Manifest** and a set of **audits** guarantee that implementations can be statically analysed for compliance.

These constraints are not optional optimisations; they are part of the structural content of the invention and are required for any implementation to be considered an instance of the present-act engine described in this defensive publication.

Okay, now write out the next subsection, making sure to, whenever there's need for it, going off of the new outline and making the updates we've discussed and we've said are going to be adding in. But if that's not the case, you're just copying from the version of the all-in-one I already gave you.

5. Context Level (CL) Ladder and Hinge Scales

5.1 Definition of the 6-Band Ladder

The **Context Level (CL) ladder** specializes the abstract context index ($n \in \mathbb{Z}$) into a concrete, six-band structure tied to physical scales. This structure is used both for interpretation

and for configuring the engine (e.g., feature resolutions, gate thresholds, and ParentGate schedules).

The six bands are:

- Level –2: **Nanoband / Biomolecular**
- Level –1: **Micron / Cellular**
- Level 0: **UGM Band / Organism Hinge**
- Level +1: **Earth-Surface Environment**
- Level +2: **Galactic Environment**
- Level +3: **Cosmic Shell / Horizon Environment**

These levels are defined as **roles** relative to a chosen Center (Level 0), not as different kinds of substance. A given physical structure may occupy different context levels under different choices of Center.

Level –2 (Nanoband / Biomolecular)

- Role: deep **inner machinery** for the present stream.
- Approximate physical scale: tens of nanometres up to on the order of a few hundred nanometres (e.g., 1–200 nm), covering molecular complexes, protein structures, and similar sub-cellular components.
- Interpretation: this band captures the fine-grained dynamical substrate (e.g., molecular and near-quantum structure) that supports the micro-dynamics of cells and local circuitry but is not directly accessible as explicit record at the Center.

Level –1 (Micron / Cellular)

- Role: **local building-block** scale for the Center.
- Approximate physical scale: fractions of a micron up to tens of microns (e.g., ~0.2–50 μm), covering whole cells, small cellular assemblies, and fine-grained tissue structure.
- Interpretation: this band serves as the immediate anatomical and functional substrate out of which the organism-scale hinge (Level 0) is built. Its details are partially compressed into effective variables at Level 0.

Level 0 (UGM Band / Organism Hinge)

- Role: **Center / Hinge** level for the present stream.

- Approximate physical scale: a narrow band around a characteristic length of order (10^{-4}) m (later identified more precisely with the Universal Geometric Mean, UGM), roughly in the ~ 0.1 – 0.2 mm range.
- Interpretation:
 - Spatially, Level 0 corresponds to the organism- or agent-scale “present pixel” at which inner and outer information are integrated into a single present-act.
 - Temporally, Level 0 is associated with a characteristic integration window (the temporal hinge) on the order of a fraction of a second; this is where many inner processes are gathered into a coherent present.

At this level, the effective inner dimension ($D(0)$) is near the hinge value (approximately 2), and the memory dimension ($D_{\text{mem}}(0)$) is close to ($D(0)$), making Level 0 the main site where inner structure and available record coincide.

Level +1 (Earth-Surface Environment)

- Role: **Immediate outer container** for the Center (local physical environment).
 - Approximate physical scale: ranges characterizing terrestrial environments (e.g., 1–100 km), including terrain, atmosphere, and local gravitational field near the surface of a planet.
 - Interpretation: this band captures the “world around the organism” in which the present stream is embedded: local environment, nearby objects, and the dominant external gravitational field.
-

Level +2 (Galactic Environment)

- Role: **Outer astrophysical container** for Level +1.
 - Approximate physical scale: length scales typical of galactic disks (e.g., kiloparsec scales), including the structure of a Milky Way–like galaxy and its rotation curve.
 - Interpretation: this band encodes the larger gravitational and structural context within which planetary systems and local environments are nested.
-

Level +3 (Cosmic Shell / Horizon Environment)

- Role: **Global container** for galaxies and large-scale structure.
- Approximate physical scale: scales associated with the observable cosmic shell (e.g., many hundreds of megaparsecs up to gigaparsec scales), capturing horizon-like or shell-like large-scale geometry.

- Interpretation: this band provides the outermost context and effective boundary conditions for the entire nested hierarchy, including the background against which galactic and planetary structures are defined.

Role-based interpretation and re-centering

The six-band ladder should always be understood as **role-based**:

- Level 0 is defined as “the Center for this vantage”. Inner levels (−1, −2) are those treated as machinery relative to that Center; outer levels (+1, +2, +3) are those treated as containers.
- If the vantage changes (e.g., from one organism to another, or to a different scale), the physical systems assigned to each band may change, but the **roles** (inner machinery, hinge, outer containers) retain their definitions and relationships.

Subsequent subsections tie these band definitions to:

- A specific **hinge scale** (Universal Geometric Mean, UGM) for Level 0, derived from the formal structure and scale bounds.
- A **temporal hinge** associated with the characteristic present-act duration.
- Evidence that key biological and physical structures cluster near the seams between these bands.

5.2 Derivation of the Universal Geometric Mean (UGM)

5.2.1 Formal Definition of UGM

The **Universal Geometric Mean (UGM)** is defined as a single characteristic spatial scale (L_{UGM}) that acts as the **global hinge** in the context ladder: the scale at which the inner and outer roles are balanced and the effective inner dimension is near the hinge value ($D_{\text{ast}} \approx 2$).

Formally, the UGM is defined as the geometric mean of a chosen **inner cutoff scale** (L_{\min}) and a chosen **outer cutoff scale** (L_{\max}):

$$\left[\begin{array}{l} L_{\text{UGM}} := \sqrt{L_{\min} \cdot L_{\max}} \end{array} \right]$$

In the application of this framework to our universe:

- (L_{\min}) is identified with a Planck-like inner length scale (ℓ_{P}), representing the lower bound at which the conventional continuum description of physics is expected to break down.

- (L_{max}) is identified with an effective outer scale (R_{obs}) , representing the characteristic size of the observable cosmic domain (e.g., a horizon or shell radius).

Thus, in physical units, the working definition is:

$$[L_{\text{UGM}} = \sqrt{\ell_{\text{P}} , R_{\text{obs}}}]$$

5.2.2 Structural Justification for Using the Geometric Mean

The use of the **geometric mean** is not an arbitrary numerical choice; it follows from structural requirements imposed by the V1 context ladder and the pure-relativity ontology.

The key requirements are:

1. Logarithmic structure of the ladder

- The formal context ladder is naturally described on a **logarithmic scale** of lengths: if (L) is a physical length, define $(x = \ln L)$.
- Moving between nested inner and outer roles corresponds to **additive** shifts in (x) , not linear shifts in (L) ; this is consistent with how fractal dimensions and scale transformations behave in V1.

2. Inner/outer role symmetry

- The ontology treats “inner” and “outer” as **roles**, not as fundamentally different substances.
- The global hinge must be **symmetric** with respect to exchanging the roles of “smallest relevant scale” and “largest relevant scale”.
- On a log scale, this symmetry is expressed by requiring the hinge to be the **midpoint** between $(\ln L_{\text{min}})$ and $(\ln L_{\text{max}})$:

$$[\ln L_{\text{UGM}} = \frac{1}{2} (\ln L_{\text{min}} + \ln L_{\text{max}})]$$

3. Uniqueness under role-exchange

- If we swap “inner” and “outer” (i.e., $(L_{\text{min}} \leftrightarrow L_{\text{max}})$), the hinge must remain unchanged:

$$[L_{\text{UGM}}(L_{\text{min}}, L_{\text{max}}) = L_{\text{UGM}}(L_{\text{max}}, L_{\text{min}})]$$

- Among simple two-argument means, the **geometric mean** is the one that:
 - is symmetric in its arguments, and
 - picks out the **logarithmic midpoint** between them.

Under these conditions, the only choice consistent with the V1 ladder structure is:

$$\begin{aligned} \ln L_{\text{UGM}} &= \frac{1}{2} (\ln L_{\min} + \ln L_{\max}) \\ L_{\text{UGM}} &= \sqrt{L_{\min} L_{\max}}. \end{aligned}$$

Other common means fail these structural tests:

- The arithmetic mean $((L_{\min} + L_{\max})/2)$ does not pick out the log-midpoint and heavily biases towards the outer scale when $(L_{\max} \gg L_{\min})$.
- The harmonic mean $(2 L_{\min} L_{\max} / (L_{\min} + L_{\max}))$ similarly fails to represent a symmetric “middle” in logarithmic space for scales spanning many orders of magnitude.

Therefore, once:

- the ladder is described on a log scale, and
- the hinge is required to be symmetric under inner/outer role exchange,

the geometric mean is structurally **forced** as the definition of the hinge scale between (L_{\min}) and (L_{\max}) .

5.2.3 Choice of Inner and Outer Bounds

Within this framework, the inner and outer bounds $((L_{\min}, L_{\max}))$ are not arbitrary:

- $(L_{\{\min\}})$ is interpreted as the **innermost physically meaningful scale** for the context ladder:
 - In the present application, this is taken to be a Planck-like scale ($\ell_{\{\text{P}\}}$), at which the classical continuum description of spacetime and fields is expected to fail.
 - This choice matches the usual inner cutoff that appears in quantum gravity considerations and is consistent with the V1 demand that there be a minimal context resolution below which “smaller” inner contexts are not meaningful in the same way.
- $(L_{\{\max\}})$ is interpreted as the **outermost context scale** relevant for the ladder:
 - In the present application, this is taken to be an effective **observable cosmic radius** ($R_{\{\text{obs}\}}$), representing the scale at which large-scale structure transitions to an approximately shell-like or horizon-like behaviour.
 - This scale appears naturally in cosmology and provides the outer bound beyond which different “versions” of the present would no longer be effectively distinguishable for the vantage in question.

The pairing $((\ell_{\text{P}}, R_{\text{obs}}))$ is therefore not used as a random numerical choice; it reflects:

- The **innermost and outermost** physically meaningful scales of the context ladder for our universe, and
- A symmetric treatment of these extremes in the pure-relativity picture (no side is privileged as “fundamental substance”).

Given this pairing, the V1 ladder and symmetry requirements imply a **single** global hinge scale (L_{UGM}) defined by their geometric mean.

5.2.4 Numerical Value and Dependence on Cosmological Inputs

Using the working choices:

- ($L_{\min} = \ell_{\text{P}}$) (a Planck-like inner cutoff), and
- ($L_{\max} = R_{\text{obs}}$) (an effective observable-universe scale),

the UGM is:

$$L_{\text{UGM}} = \sqrt{\ell_{\text{P}} \cdot R_{\text{obs}}}.$$

For typical values of (ℓ_{P}) and (R_{obs}) used in contemporary cosmology, this expression yields a characteristic length in the approximate range:

- ($L_{\text{UGM}} \sim 10^{-4} \text{ m}$),

i.e., on the order of **0.1 mm**, consistent with a **UGM band** of roughly (0.1)–(0.2) mm once uncertainties in cosmological parameters and choice of effective outer radius are taken into account.

Important points:

- The exact numerical value of (L_{UGM}) depends on the chosen (R_{obs}) (e.g., whether one uses a particle horizon, event horizon, or an effective shell radius) and on the adopted value of (ℓ_{P}).
- The framework therefore treats the **band** around (L_{UGM}) (e.g., (0.1)–(0.2) mm) as the relevant hinge interval rather than a single exact number.
- Subsequent sections document that this band aligns with:
 - empirically observed clustering of structural features (e.g., in biology and sensory resolution), and
 - the role of Level 0 as the **Organism Hinge** in the CL ladder.

In summary:

- The definition of (L_{UGM}) as a geometric mean is structurally dictated by the V1 context ladder and pure-relativity constraints.
- The specific pairing $((\ell_{\text{P}}, R_{\text{obs}}))$ is justified by the identification of inner and outer physical cutoffs.
- The resulting hinge band around (~ 0.1) mm is then used as the Level 0 spatial hinge in both the Context Level Framework and the engine configuration.

5.3 Temporal Hinge (T^*) and Relation to Budgets and Context

5.3.1 Definition of the Temporal Hinge

In parallel with the spatial hinge (L_{UGM}) , the framework introduces a **temporal hinge** (T^*). This is defined as the characteristic integration window for a **single present-act** at the Center (Level 0):

- (T^*) is the timescale over which many inner processes (e.g., neural, cellular, and micro-physical events in bands -2 and -1) are integrated and presented as **one** coherent present-act at Level 0.
- It is not the minimal tick size of the engine; rather, it is an emergent, band-specific “specious present” unit associated with the hinge.

At the conceptual level, (T^*) is the **temporal analogue** of (L_{UGM}) :

- (L_{UGM}) acts as a “present pixel” in space for Level 0.
- (T^*) acts as a “present pixel” in time for Level 0.

In quantitative applications to human-scale systems, (T^*) lies in a band of order (~ 0.1) s, consistent with:

- psychophysical estimates of the minimum integration time needed for a coherent perceptual moment, and
- conduction and integration delays in typical nervous systems.

The framework treats this as a **hinge band** around (T^*), rather than a single exact value.

5.3.2 Relation to Typed Budgets and Base Units

In the discrete invariant-interval construction, the framework defines typed budgets:

- $(\Delta \tau = N_{\tau}, \tau^*)$ (proper-time-like budget),
- $(\Delta x = N_x, x^*)$ (space-like budget),
- $(\Delta t = N_t, t^*)$ (time-like budget),

with unit scales (τ^*, x^*, t^*) constrained by:

$$\left[\Delta t^2 = \Delta \tau^2 + \frac{\{\Delta x^2\}}{\{c^2\}}, \right]$$

for a characteristic speed (c).

At the level of **single-step micro-updates**, one can choose base units such that:

- (x_*) is the spatial step associated with a minimal configuration change,
- (t_*) is the time step associated with that minimal change, and
- the relation between them at the “lightlike” limit $((\Delta \tau = 0))$ yields:

$$\left[t_* = \frac{\{x_*\}}{\{c\}}. \right]$$

In this scheme:

- (T^*) is **not** identified with (t_*) ; instead, it corresponds to the accumulation of many base ticks:

$$\left[T^* = N_T, t_*, \right]$$

where (N_T) is a large integer representing how many micro-updates are integrated into one Level 0 present-act.

Similarly, (L_{UGM}) is not forced to equal (x_*) ; rather, it anchors the **spatial scale** at which Level 0 acts are centered. A natural choice in many applications is:

- pick (x_*) small compared to (L_{UGM}) , and
- treat (L_{UGM}) as defining a band of many micro spatial steps that together constitute one “present pixel” at the hinge.

Thus, the relationship among (T^*) , (L_{UGM}) , (x_*) , (t_*) , and (c) can be summarized schematically as:

- (c) is fixed by the discrete invariant interval at the micro level $((x_*/t_* = c)$ in the appropriate limit).
- (L_{UGM}) and (T^*) define the hinge scales as **aggregates** over many micro steps:

$$\left[\begin{aligned} L_{\text{UGM}} &\approx N_L, x_*, \\ T^* &\approx N_T, t_*, \end{aligned} \right]$$

for integers (N_L, N_T) determined by how many micro updates the hinge is taken to integrate.

The precise values of (N_L) and (N_T) are implementation choices constrained by CL evidence and by the requirement that the hinge behave as a coherent present-act.

5.3.3 Temporal Hinge Conditions at Level 0

From the standpoint of the context ladder and the V1 formalism, the temporal hinge (T^*) at Level 0 satisfies:

- **Integration condition:**
Over an interval of length (T^*), a large number of micro events at levels -2 and -1 are integrated into one coherent Level 0 act; micro fluctuations within that window are not individually presented as separate acts at the Center.
- **Coherence condition:**
Changes in Level 0 qualia and world records over intervals significantly shorter than (T^*) are not treated as distinct “presents” by the engine at that level; they are part of the same act. Conversely, changes over intervals significantly longer than (T^*) must be represented by sequences of distinct present-acts.
- **Band-anchoring condition:**
In the CL mapping, Level 0 is assigned a specific temporal hinge band centered on (T^*), while inner and outer bands have their own characteristic timescales (e.g., faster dynamics at $-2/-1$, slower environmental changes at $+1$ and above). These scale differences are used when defining temporal constraints in the Θ gate and related feasibility conditions.

These conditions ensure that Level 0 behaves as the **temporal center** of the ladder in the same way that (L_{UGM}) makes it the spatial center: it is the level at which the present-act is both defined and experienced, with inner machinery and outer containers feeding into and out of that hinge.

5.3.4 Empirical Range and Interpretation

When the framework is applied to human or animal nervous systems, empirical considerations (psychophysical data, conduction velocities, central integration times) suggest:

- a temporal hinge (T^*) in the approximate range of **tens to a few hundred milliseconds**, often summarized as (~ 0.1) s, and
- consistency between this range and:
 - conduction delays for signals to reach central processing hubs,
 - known integration windows for perception (e.g., motion, flicker fusion, temporal binding),
 - characteristic time windows used in behavioural and cognitive tasks.

The framework does not insist on an exact number; instead, it treats the **hinge band** around (T^*) as the temporal present for Level 0. Within this band, micro events are integrated into a single act; outside this band, distinct acts must be introduced.

In summary:

- The temporal hinge (T^*) is the Level 0 integration window for present-acts.
- It is structurally related to the typed budgets and the invariant interval via the base units $((x_*, t_*))$ and the speed (c), but it arises as an **aggregate timescale**, not as a primitive tick.
- Empirical evidence places (T^*) in a specific band (on the order of (~ 0.1) s for human-scale systems), which is used to configure the engine and CL constraints at the center of the ladder.

5.4 CL Evidence Summary (Non-Narrative)

5.4.1 Inner-Band Probes ($-2, -1, 0$)

The Context Level Framework is supported by a set of **inner-band probes** that collect and analyse characteristic length scales from biological and physical systems and compare them to the predicted band structure and seams:

- **Probe $-2/-1$ (Nanoband \rightarrow Micron seam)**
 - Data sets covering biomolecular structures and sub-cellular features in the ~ 1 – 200 nm and ~ 0.2 – 50 μm ranges.
 - Analysis of fractal-dimension plateaus and geometric-mean relations between characteristic nanoscales and cell-scale structures.
 - Evidence that many “sub-cellular infrastructure” scales cluster near the geometric-mean pivots identified for the $-2 \leftrightarrow -1$ seam.
- **Probe $-1/0$ (Micron \rightarrow UGM seam)**
 - Data sets covering whole-cell diameters, small multicellular structures, and organism-level microstructures near the predicted UGM band (~ 0.1 – 0.2 mm).
 - Examination of how cell-scale and small-organism-scale characteristic sizes relate via geometric means to the UGM scale.
 - Evidence of clustering of characteristic sizes near the $-1 \leftrightarrow 0$ seam, consistent with the interpretation of Level 0 as the first “as-one-with-parts” hinge level.
- **Probe 0 (UGM band)**
 - Targeted analysis of structures in the ~ 0.1 – 0.2 mm range (biological and other).
 - Fractal-dimension and scale-clustering studies around this band, including tests for whether this range behaves as a pivot or plateau in multi-scale distributions.
 - Evidence that a disproportionate number of characteristic scales (across diverse systems) fall near the UGM band, supporting its role as a global hinge in the ladder.

These probes are documented in the attached CL and probe reports (e.g., -2 CL, -1 CL, 0 CL, and related probe summaries) and provide the primary empirical support for identifying -2 , -1 , and 0 as distinct inner/mid bands with a hinge at Level 0.

5.4.2 Outer-Band Probes (+1, +2, +3)

Outer-band probes examine how characteristic scales of environments and astrophysical structures align with the +1, +2, and +3 roles:

- **Probe +1 (Earth-surface band)**
 - Collection of characteristic terrestrial scales (e.g., typical vertical and horizontal scales of environmental features, atmospheric scales, and other Earth-surface phenomena) in a band roughly corresponding to 1–100 km.
 - Analysis of whether these scales form plateaus or clustering near the predicted +1 band boundaries.
 - Evidence that the Level +1 band acts as a natural “container scale” for Level 0 organisms in a wide variety of environments.
- **Probe +2 (Galactic band)**
 - Data sets on galactic disk sizes, typical star–star separations, and related galactic-structure measures.
 - Specific attention to Milky Way–like disk sizes and their relation to geometric means of smaller and larger context scales.
 - Evidence of a **fractal geometric-mean pivot** at a Milky Way–scale band, consistent with treating +2 as the galactic context level and with the later gravity/lensing tests that find “activation” effects near this scale.
- **Probe +3 (Cosmic shell band)**
 - Use of large-scale cosmological scales (e.g., horizon-like or shell-like structures, correlation lengths in large-scale structure) to define and test the +3 band.
 - Examination of whether characteristic cosmic scales cluster near the predicted outer seam and whether they can be coherently interpreted as an effective “cosmic shell” container.
 - Evidence that the chosen outer scale (R_{obs}) used in the UGM construction is consistent with where structural features in cosmology change character (e.g., transition to shell-like/horizon behaviour).

These probes, documented in the +1, +2, and +3 CL analysis documents, support the assignment of +1, +2, and +3 as successively larger containers in the ladder, each associated with distinctive clustering and plateau behaviour in real data.

5.4.3 Cross-Band Hinge Alignments

Beyond band-wise clustering, the CL evidence includes **cross-band hinge tests** that look at geometric-mean relationships and alignments between different context levels:

- **Planck–Universe geometric mean → UGM band (Level 0)**
 - As described in Section 5.2, the geometric mean of an inner Planck-like scale and an outer observable-universe scale lands in the ~ 0.1 – 0.2 mm range.
 - CL probes test whether characteristic scales across domains (e.g., biological structures, sensory resolution thresholds) show enhanced clustering near this range, consistent with Level 0 as a global hinge.

- **UGM–Earth geometric mean → CNS/organism size band**
 - CL probes examine whether the geometric mean between UGM and Earth-surface length scales corresponds to a band in which central nervous systems and organism sizes cluster.
 - Evidence of such clustering supports the idea that Level 0 (UGM band) and Level +1 (Earth band) jointly define a “window” in which organisms with global present-acts are most naturally situated.
- **Other cross-band GMs (e.g., inner/outer → intermediate band)**
 - Similar tests are performed for other pairs of bands to see whether geometric-mean lengths align with observed clusters of characteristic scales in relevant domains.
 - Where alignment is found, it reinforces the interpretation of context seams as natural pivot points in the scale hierarchy.

The CL evidence does not claim that every scale aligns perfectly with these geometric means; rather, it identifies **statistically non-trivial clustering** near predicted seams and hinge scales across multiple, heterogeneous datasets.

5.4.4 Limitations and Status of CL Evidence

The CL evidence programme, as recorded in the attached CL and probe documents, has the following status and limitations:

- It provides **qualitative and semi-quantitative support** for:
 - the existence of distinct scale bands that behave differently (inner machinery, hinge, containers), and
 - the presence of hinge-like behaviour near certain predicted scales (e.g., ~0.1–0.2 mm for Level 0, Milky Way-like scales for Level +2, and a cosmic shell scale for Level +3).
- It does **not** yet constitute:
 - a complete statistical model of all relevant scale distributions across all domains, or
 - a full proof that the context ladder is uniquely determined by the data.
- The evidence is strongest where:
 - multiple independent datasets (e.g., from different biological systems or different astrophysical observations) show clustering near the same predicted seams; and
 - those seams can be tied back to geometric-mean constructions grounded in the V1 context ladder (e.g., Planck–Universe → UGM).
- The programme explicitly marks:
 - which probes show strong alignment,
 - which show weaker or ambiguous patterns, and
 - where additional data or improved analysis would be required to refine or challenge the CL assignments.

In this defensive publication, the role of the CL evidence summary is to:

- State that there exists a structured set of CL probes and analyses supporting the six-band ladder and hinge scales, and
- Indicate that these probes are documented in detail in the attached CL and probe volumes, which are incorporated here by reference.

6. Gravity as Feasibility Geometry and Gravitational Amplitude χ

6.1 Gravity in V1: Hinge and Pivot Profile

In the formal core (V1), **gravity** is not introduced as curvature of a pre-given spacetime manifold. Instead, it appears as a specific way in which the **hinge structure** and **pivot profile** bias which histories are feasible across the context ladder.

The key ingredients are:

- the **context ladder** ($\{\mathcal{L}_n\}$) with dimension profile ($D(n)$),
- the **pivot function** ($g(D)$), normalized at a hinge dimension ($D_{\text{ast}} \approx 2$),
- and the **hinge projection** (Hinge Projection Operator), which localizes context-time action to a thin boundary and then expresses it as a field-like influence in a neighbourhood.

Together, these define how **feasibility** varies as a function of position in the ladder and around containers (e.g., planets, stars, galaxies), and this variation is what will later appear as “gravitational” behaviour.

Hinge-based view of gravity

At the V1 level:

- Each context level (\mathcal{L}_n) has an effective inner dimension ($D(n)$) and a corresponding pivot value ($g(D(n))$).
- Levels where ($D(n)$) is near the hinge dimension (D_{ast}) (with ($g(D(n))$) near its normalized reference) are **hinge-like**: they behave as natural **surfaces** or **shells** where inner and outer structure are in balance.
- Levels far from the hinge (either too sparse or too dense) behave more like bulk interior machinery or distant outer containers.

Gravity is interpreted as arising from:

- The way **hinge-like layers** in this ladder act as privileged surfaces of feasibility,
- And the way **containers** (e.g., Level +1 Earth, Level +2 galactic disk) inherit hinge-like properties at their boundaries and exert feasibility gradients inwards on admissible histories.

In this picture, a “gravitational field” is not a separate entity; it is the emergent pattern of **how easy or hard it is** for certain histories (configurations and trajectories) to persist or occur, given the pivot profile and hinge structure of the nested contexts.

Pivot profile and container bias

Given the dimension profile ($D(n)$) and pivot function ($g(D)$), one can define a **pivot profile over the ladder**:

- At each level (n), the value ($g(D(n))$) measures how hinge-like (or non-hinge-like) the inner structure is.
- When a level plays the role of a **container** (e.g., a planetary surface, a galactic disk), its hinge-like boundary acts as a natural **bias surface**:
 - Histories that respect that boundary (e.g., staying near the surface, following certain shell-like paths) are structurally favoured.
 - Histories that deviate strongly from the container’s hinge geometry (e.g., trying to “escape” without sufficient feasibility) are structurally disfavoured.

This bias is expressed formally as **differences in feasibility** across the ladder and around the container’s hinge surface, rather than as an externally imposed force field.

Hinge Projection and field-like behaviour

The **Hinge Projection Operator** (historically nicknamed “division-by-zero operator”) has a specific function in this context:

- It **localizes** context-time action to the hinge region of a container, effectively concentrating the relevant dynamics on a thin, near-2D shell.
- It then **promotes** this localized action to a 4D-like field description in a neighbourhood of that shell, so that feasibility effects can be described in a smooth-looking way at larger scales.

At the formal level, this yields:

- A field-like description of feasibility gradients around hinge-like surfaces (e.g., near the Earth’s surface, within a galactic disk).
- The ability to talk about “effective curvature” or “potential wells” as **summaries** of how the underlying present-act dynamics and pivot profile bias histories in that region.

No literal division by zero occurs in this process; the operator is a structural mapping in the carrier and interval algebra, not a numerical $1/0$.

From V1 structure to gravity-like predictions

Because:

- The context ladder identifies specific hinge-like levels and seams,
- The pivot profile ($g(D(n))$) singles out natural surfaces where inner and outer structure balance, and
- The hinge projection expresses these as field-like feasibility gradients,

the formal framework provides:

- A way to **predict** which scales and surfaces should exhibit gravity-like behaviour (e.g., Milky Way-scale disks, planetary surfaces, cosmic shells).
- A structural mechanism by which redshift, time dilation, deflection, and horizon-like behaviour can arise, once the V1 hinge/pivot structure is realized in a concrete engine (V2) as **feasibility geometry**.

The next subsections describe:

- How this V1 structure is implemented in V2 as **ParentGate and shell schedules**,
- How a **gravitational amplitude χ** is defined from context-ladder scales and pivot geometry, and
- How χ compares to standard GR amplitudes, including an explicit discussion of the observed factor-of-order-unity discrepancy.

6.2 V2.1 Gravity Implementation: ParentGate and Shells

In the engine (V2.1), gravity is implemented as a **feasibility geometry** via a specific gate called **ParentGate**. ParentGate is the only gate that is allowed to depend directly on large-scale radial structure (shells) and on the gravitational amplitude χ (defined in the next subsection). All other gates remain purely local and non-gravitational.

6.2.1 Shell Structure Around Containers

For each **container** (e.g., a planet at Level +1, a galactic disk at Level +2), the engine defines a family of **radial shells**:

- Space around the container is discretized into a finite sequence of concentric shells:

$$\begin{bmatrix} S_1, S_2, \dots, S_N, \end{bmatrix}$$
 ordered by increasing radial distance (r) from the container's center.
- The manifest specifies:
 - The total number of shells (N) for each container type.
 - The radial boundaries of each shell (in whatever coordinate system is appropriate for the frame).
 - A **base strictness level** or **feasibility weight** associated with each shell index.

Shell structure is:

- **Band-aware:** shells are tagged with which context level they belong to (e.g., shells for a Level +1 container vs shells for a Level +2 container).
- **Container-specific:** different containers (e.g., Earth vs star vs galaxy) have their own shell sets, but all obey the same structural rules declared in the manifest.

6.2.2 ParentGate as a Radial Feasibility Gate

ParentGate operates on candidate transitions that involve movement relative to a container. For any candidate ($c = (w_{k+1}, q_k)$) that has passed the non-gravity gates:

- The engine determines:
 - Which container(s) the candidate is associated with (e.g., Earth, a specific galaxy).
 - The shell index (s) in which the relevant part of the configuration lies at tick (k).
 - The shell index (s') in which it would lie at tick ($k+1$).

Using these indices, ParentGate:

- Consults the manifest for the **base strictness schedule**:

$$[\sigma_{\text{base}}(1), \sigma_{\text{base}}(2), \dots, \sigma_{\text{base}}(N),]$$
 where each ($\sigma_{\text{base}}(s)$) is an integer or small ordinal representing how “hard” it is for histories to occupy shell (S_s).
- Adjusts this base schedule by the gravitational amplitude χ (described in Section 6.3) to obtain an **effective strictness schedule**:

$$[\sigma_{\text{eff}}(s) = f(\sigma_{\text{base}}(s), \chi),]$$
 where (f) is a manifest-declared function (e.g., integer scaling or a simple rational reweighting rule).

ParentGate then evaluates the candidate based on:

- The shells it moves through or occupies (from (s) to (s')), and
- The effective strictness levels ($\sigma_{\text{eff}}(s)$) and ($\sigma_{\text{eff}}(s')$).

A candidate is **rejected** by ParentGate if, for example:

- It attempts to occupy or cross into shells whose strictness is above the allowed threshold for that kind of move (e.g., moving “uphill” into a much stricter shell without appropriate compensating changes).
- It implies a pattern of motion across shells that is inconsistent with the manifest-declared feasibility rules (e.g., a large radial jump that would violate the no-skip/locality conditions when expressed in shell indices).

A candidate is **accepted** with respect to ParentGate if:

- Its shell usage and radial movement are consistent with the effective strictness schedule, and
- Any additional container-specific constraints (e.g., horizon-like cutoffs) are satisfied.

ParentGate always outputs a **boolean pass/fail**. It does not itself assign probabilities or continuous scores.

6.2.3 Monotone Strictness and Horizon Templates

The manifest must ensure that the effective strictness schedule ($\sigma_{\text{eff}}(s)$):

- Is **monotone non-decreasing** with increasing radius for the relevant configuration (or follows a monotone pattern appropriate to the container, as specified).
- Encodes horizon-like or shell-like behaviour where required.

Typical patterns include:

- **Planetary-type containers (Level +1):**
 - Lower strictness near preferred orbital/near-surface regions (where “stable” histories are more feasible).
 - Increasing strictness both deep inside (below certain shells) and far outside (beyond escape regions), reflecting the difficulty of maintaining histories that leave the near-surface region.
- **Galactic-type containers (Level +2):**
 - Strictness patterns that capture disk-like regions where histories naturally circulate (e.g., near preferred radii), with stronger constraints near the core and far outer regions.
- **Cosmic shell containers (Level +3):**
 - Schedules that represent an effective outer boundary or horizon, beyond which histories are heavily suppressed, corresponding to the role of Level +3 as a cosmic shell.

These schedules allow the engine to reproduce qualitative features of gravitational wells and horizons:

- Regions where histories are “trapped” (easier to remain than to leave).
- Regions where certain paths (e.g., radial escapes) are structurally disfavoured.
- Regions where deflection-like effects emerge because “straight-line” paths in world coordinates are not the most feasible under ($\sigma_{\text{eff}}(s)$).

6.2.4 Rotation Invariance and Shell Symmetry

To match the observed isotropy of gravitational fields around approximately spherical or shell-like containers, ParentGate must respect **rotation invariance** in expectation:

- The strictness schedule depends on **shell index (s)** only, not on angular coordinates.
- For a given shell (S_s), candidates at different angular positions but the same radius are treated identically by ParentGate, provided their other tags (e.g., mass distribution, context) are equivalent.

In discrete engine implementations:

- Local anisotropies may exist at fine resolution (e.g., due to discretization or specific manifest choices), but audit tools must verify that, on appropriate coarse-grained scales, ParentGate behaves **effectively isotropically** around containers that are intended to be symmetric.

This symmetry is crucial for ensuring that:

- Emergent gravitational effects (e.g., deflection, redshift, time delay) do not depend on arbitrary grid orientation or coordinate artefacts, and
- The engine’s outputs match the qualitative symmetries expected of weak-field gravity in the relevant contexts.

6.2.5 Separation from Non-Gravity Gates and Role of χ

ParentGate is **structurally and functionally separated** from the non-gravity feasibility gates:

- Non-gravity gates (Θ , κ , structural, CRA-like) enforce:
 - temporal coherence,
 - granularity and persistence,
 - local structural integrity, and
 - context-resolved admissibility,

without any direct dependence on shells or gravitational amplitude.

- ParentGate alone:
 - Uses shell indices and container roles,
 - Applies the effective strictness schedule ($\sigma_{\text{eff}}(s)$),
 - And depends on the gravitational amplitude χ (next subsection) to set the **overall strength** of feasibility gradients around containers.

The **amplitude χ** acts as a single, container-level parameter:

- It scales or otherwise modulates the base strictness schedule for a given container (e.g., Earth, a galaxy).
- It sets how “deep” or “shallow” the effective feasibility well is around that container.
- It is **not** tuned per observable; the same χ must simultaneously govern:
 - deflection of light/rays,
 - time delays (Shapiro-like effects), and
 - redshift/time-dilation patterns,

in the relevant engine simulations.

In summary:

- V1 provides a hinge and pivot structure that motivates container-based feasibility gradients.
- V2.1 realizes this as ParentGate operating on shells with a strictness schedule modulated by χ .
- All gravity-like behaviour in the engine arises from how ParentGate filters histories in shell space, in combination with the non-gravity constraints and the invariant interval.

6.3 Definition and Derivation of Gravitational Amplitude χ

6.3.1 Formal Definition of χ

In the engine, the **gravitational amplitude** χ is a single, **dimensionless** parameter that sets the overall strength of the feasibility gradients applied by ParentGate around a given +1 container (e.g., Earth). It is constructed from three context-ladder scales, the “Hinge Triple”:

- (L_{UGM}) – the spatial hinge / “present pixel” at Level 0 (UGM band),
- (R_{oplus}) – the characteristic radius of the +1 container (Earth band),
- (R_{obs}) – the effective radius of the outermost container (cosmic shell / Level +3).

The canonical definition used in the v0.9 bundle is:

$$\chi \equiv \frac{R_{\text{oplus}}^2}{L_{\text{UGM}} \cdot R_{\text{obs}}}$$

This expression:

- is **dimensionless** ($\text{length}^2 / (\text{length} \times \text{length})$),
- depends only on the **three ladder scales** (L_{UGM} , R_{oplus} , R_{obs}), and
- has the required monotonicity properties:
 - χ increases if (R_{oplus}) increases (a larger container is “deeper”),
 - χ increases if (L_{UGM}) decreases (finer hinge resolution makes the container effectively deeper),
 - χ increases if (R_{obs}) decreases (a smaller cosmic container makes our local container more nested).

This χ is then used to modulate the strictness schedule of ParentGate for the Earth container (and, by extension, for other +1 containers in appropriately scaled contexts), without inserting a separate mass parameter or Newton’s constant (G) into the engine’s control path.

6.3.2 Structural Derivation from Context Ratios and Pivot Geometry

The choice of χ is constrained by the **structure** of the context ladder and the pure-relativity picture. It is not intended as a numerological fit, but as the simplest monomial of lengths that satisfies a small set of structural requirements.

The requirements are:

1. Use only context-ladder length scales

- Gravity strength should be encoded purely in terms of:
 - the Level 0 hinge scale ($L_{\{\text{UGM}\}}$),
 - the +1 container size (R_{\oplus}), and
 - the outer container/shell size ($R_{\{\text{obs}\}}$).
- No masses, densities, or external constants (like (G)) are allowed as primitive inputs in the control path; they must, if they appear at all, be emergent from how χ and other structural parameters interact.

2. Dimensionless and scale-ratio based

- χ must be **dimensionless**, consistent with the theory's emphasis on relativity of scale rather than absolute magnitudes.
- It must be built from **ratios of lengths**, reflecting how many times a smaller scale fits inside a larger one.

3. Encode “0↔+1 depth” and “+1↔+3 nesting”

Conceptually, χ should measure:

- How many Level 0 “pixels” fit across the +1 container:
[
 $\frac{R_{\oplus}}{L_{\{\text{UGM}\}}}$
]
(how many 0-scale grains make up the container's radius).
- How nested that container is inside the outer shell:
[
 $\frac{R_{\oplus}}{R_{\{\text{obs}\}}}$
]
(relative depth of +1 inside +3, as seen from the Center).

Both of these scale ratios should influence χ : if either the container shrinks (smaller (R_{\oplus})) or the outer shell expands (larger ($R_{\{\text{obs}\}}$)), the effective gravitational amplitude at the Center should decrease.

4. Monotonicity and minimal complexity

- χ should **increase** when:
 - (R_{\oplus}) increases (a larger container feels “deeper” at its surface),
 - ($L_{\{\text{UGM}\}}$) decreases (finer hinge resolution increases sensitivity to the container),
 - ($R_{\{\text{obs}\}}$) decreases (a smaller cosmic container makes our container more nested).

- Among simple monomials of $(L_{\{\text{UGM}\}}, R_{\text{oplus}}, R_{\{\text{obs}\}})$ with these monotonicities, the theory prefers the **simplest** one consistent with the above constraints.

Given these requirements, a natural dimensionless candidate built from the two intuitive scale ratios is:

- $(R_{\text{oplus}}/L_{\{\text{UGM}\}})$ (many 0-scale pixels across the container radius), and
- $(R_{\text{oplus}}/R_{\{\text{obs}\}})$ (how deep the container sits inside the outer shell).

Multiplying them:

$$\left[\frac{R_{\text{oplus}}}{L_{\{\text{UGM}\}}} \times \frac{R_{\text{oplus}}}{R_{\{\text{obs}\}}} \right] = \frac{R_{\text{oplus}}^2}{L_{\{\text{UGM}\}} R_{\{\text{obs}\}}},$$

yields exactly the χ used in the v0.9 bundle.

This expression:

- is the **minimal monomial** that simultaneously:
 - uses only the three ladder scales,
 - yields the desired monotonicities, and
 - encodes both the $0 \leftrightarrow +1$ grain count and the $+1 \leftrightarrow +3$ nesting depth,
- and it does so **without** introducing any extra dimensionless factors (e.g., (4π) , $1/2$, etc.) that would amount to a further, unexplained choice.

Other monomials either:

- violate one of the monotonicity conditions,
- de-emphasize one of the two conceptual components (e.g., only counting grains across Earth but ignoring cosmic nesting), or
- introduce unnecessary complexity without providing a clear structural advantage.

For that reason, the framework adopts:

$$\left[\chi = \frac{R_{\text{oplus}}^2}{L_{\{\text{UGM}\}} R_{\{\text{obs}\}}} \right]$$

as the **canonical gravitational amplitude** associated with the Earth container at v0.9.

6.3.3 Expected Range and Sensitivity

Using working values appropriate to the present universe:

- (L_{UGM}) in the hinge band around ($\sim 10^{-4}$) m (on the order of 0.1–0.2 mm),
- ($R_{\text{oplus}} \approx 6.37 \times 10^6$) m (Earth radius),
- (R_{obs}) in a band around the effective observable-universe radius (cosmic shell scale),

the above expression yields a χ in the approximate range:

$$[\chi \sim 10^{-9} \text{ to } 10^{-10},]$$

with the specific value used in the v0.9 engine and simulations being approximately:

$$[\chi_{\text{engine}} \approx 7.7 \times 10^{-10}.]$$

Key points about this value:

- It is **entirely determined** from the three ladder scales; no independent tuning to match any particular gravitational observable (e.g., specific deflection or redshift measurements) is performed.
- Its precise value depends on:
 - The chosen physical value of (L_{UGM}) within its hinge band,
 - The adopted effective cosmic radius (R_{obs}) (e.g., particle horizon vs event horizon vs shell scale),
 - The reference Earth radius used (e.g., equatorial vs mean radius).

Because each of these has some uncertainty, the framework treats χ as belonging to a **narrow amplitude family** rather than as a single exact number. The v0.9 choice ($\approx (7.7 \times 10^{-10})$) is one representative member of this family.

In subsequent subsections:

- χ is compared to the standard GR metric potential amplitude (e.g., (r_s/R_{oplus})) and the observed factor-of-order-unity discrepancy (~ 1.8) is explicitly discussed, and
- The performance of this χ -family in reproducing deflection, delay, and redshift patterns in the engine simulations is summarized, to show how much of familiar gravitational behaviour can already be recovered from this structurally derived amplitude.

6.4 Comparison to GR Metric Amplitude and Factor ≈ 1.8 Discrepancy

6.4.1 Standard GR Reference

In weak-field general relativity (GR), the dimensionless strength of the gravitational potential at the surface of a spherical mass (M) with radius (R) is typically characterized by:

$$\left[\Phi_{\text{GR}} \sim \frac{r_s}{R}, \right]$$

where:

- ($r_s = \frac{2 G M}{c^2}$) is the Schwarzschild radius associated with the mass (M),
- (R) is the physical radius at which the potential is evaluated (for Earth, ($R = R_{\oplus}$)).

For Earth, using standard values of (G, M_{\oplus} , c,) and (R_{\oplus}), one obtains:

$$\left[\Phi_{\text{GR}} \approx 1.4 \times 10^{-9} \right]$$

as the order-of-magnitude dimensionless metric potential at the surface.

This quantity is not a directly observable number by itself, but it sets the scale for:

- gravitational time dilation factors near Earth's surface,
- characteristic deflection angles in weak lensing regimes (when combined with geometry),
- and other small, dimensionless gravitational parameters in the weak-field limit.

6.4.2 Numerical Comparison and Discrepancy

From the CL ladder and hinge construction, the framework defines a gravitational amplitude:

$$\left[\chi = \frac{R_{\oplus}^2}{L_{\text{UGM}}}, R_{\text{obs}} \right],$$

which, for the working values used in the v0.9 bundle, yields:

$$\left[\chi_{\text{engine}} \approx 7.7 \times 10^{-10}. \right]$$

Comparing this to the GR reference amplitude:

- GR metric potential amplitude at Earth’s surface:
[
 $\Phi_{\text{GR}} \approx 1.4 \times 10^{-9}$,
]
- Engine gravitational amplitude:
[
 $\chi_{\text{engine}} \approx 7.7 \times 10^{-10}$.
]

The ratio is:

$$\left[\frac{\Phi_{\text{GR}}}{\chi_{\text{engine}}} \approx \frac{1.4 \times 10^{-9}}{7.7 \times 10^{-10}} \approx 1.8. \right]$$

Thus:

- The **order of magnitude** of χ matches that of the GR metric amplitude at Earth’s surface.
- There is an explicit, **order-unity discrepancy** by a factor of approximately **1.8**.

This discrepancy is not hidden; it is part of the on-record comparison.

6.4.3 Interpretation and Status

Within this framework, the factor ≈ 1.8 difference between χ_{engine} and Φ_{GR} is interpreted as:

- **Not** a failure of structure—since χ was never tuned to match Φ_{GR} exactly,
- **But also not** something to be ignored—since the framework aspires to produce quantitative matches where possible.

Several points are important for interpreting this gap:

1. **χ is structurally constrained, not fitted to GR**
 - χ is constructed solely from the three context-ladder scales (L_{UGM} , R_{oplus} , R_{obs}) and simple structural requirements (dimensionless, correct monotonicities, inner/outer nesting).
 - No mass, density, or Newton’s constant (G) is inserted into its definition.
 - No extra order-one factors (such as 2, (4π) , $1/2$, etc.) are added to force a match to GR.
 - The resulting amplitude is therefore a **prediction** of the framework’s ladder-based construction, not a post-hoc fit.

2. Approximation layers and input uncertainties

- The effective outer scale (R_{obs}) is itself model-dependent (choice of horizon/shell definition, cosmological parameters). Small changes in this choice can shift χ by order-one factors.
- The UGM hinge scale (L_{UGM}) is tied to the geometric mean of inner and outer scales, but is used as a **band** (e.g., $(\sim 0.1) - (0.2)$ mm), not as a single exact number. Different consistent choices within this band also shift χ .
- Earth is not a perfect thin spherical container with uniform density; the framework currently treats it as such at the level of χ , leaving detailed density profiles to be absorbed into how ParentGate schedules are calibrated around (R_{oplus}).

Taken together, these approximations and input uncertainties easily allow for order-unity shifts in the effective amplitude.

3. Single χ used across multiple gravitational observables

- The same χ is used to modulate ParentGate's strictness schedule for all gravity-like effects in the engine:
 - deflection patterns in lensing scenes,
 - gravitational redshift/time-delay analogues,
 - orbit-like confinement and horizon-like behaviour.
- There is **no separate χ per observable**; one amplitude must simultaneously support all three classes of phenomena in simulations.
- In practice, with $\chi \approx (7.7 \times 10^{-10})$, the engine reproduces:
 - deflection angles of the correct order of magnitude,
 - time-delay and redshift-like behaviours in the right ballpark,
 - and horizon/proximity effects qualitatively consistent with weak-field expectations, without observable-specific tuning.

4. Status as a refinement target

- In the current v0.9 state, the factor ≈ 1.8 is treated as:
 - a **quantitative tension** to be refined by improved modelling of:
 - the context mapping from ladder to physical scales,
 - the detailed shape of the pivot profile ($g(D)$), and
 - container-specific structure (e.g., more realistic Earth and cosmic shells),
 - rather than as an indicator that the χ -construction is fundamentally wrong.
- Future revisions may:
 - adjust the mapping between abstract ladder indices and physical scales,
 - refine the Hinge Triple or introduce additional structural constraints that update the χ -formula,
 - or show that the discrepancy is absorbed once higher-order GR effects or more precise data are integrated.

In short:

- χ and (Φ_{GR}) agree in order of magnitude.

- Their ratio (~ 1.8) is explicitly acknowledged and is **not** removed by arbitrary tuning.
- The current framework treats this as a **first-pass structural prediction** that is close but not exact, and it identifies this gap as a clear target for future theoretical and empirical refinement, rather than as a reason to abandon the ladder-based χ construction.

6.5 Gravitational Budgets and Engine Constraints

This subsection collects the explicit constraints that tie **gravity-as-feasibility** to the **typed budgets** and to the core engine axioms, ensuring that gravitational behaviour in the engine is consistent with the discrete invariant interval and with causal structure.

6.5.1 Budgets as Read-Out Variables (Not Control Inputs)

In the engine, the typed budgets

- $(\Delta \tau)$ (inner-time / proper-time-like),
- (Δt) (outer-time),
- (Δx) (outer-space),

are treated as **read-out variables** derived from discrete counts associated with an accepted act:

- After a transition ($k \rightarrow k+1$) has been fully selected (via selectors, hinge equality, gates, ratio-lex, fewest-acts, and, if needed, PF/Born ties-only), the engine computes:
 - an integer count of inner-time steps (e.g., (p_k)),
 - an integer count of outer-time steps (e.g., (m_k)),
 - an integer count of spatial steps (e.g., (n_k)),

and converts them into budgets via fixed constants:

$$\begin{aligned} &[\\ &|\Delta x_k| = n_k, \ell_{\mathrm{H}}, \quad \\ &|\Delta t_k| = m_k, T^*, \quad \\ &|\Delta \tau_k| = p_k, \tau_0, \\ &] \end{aligned}$$

where (ℓ_{H}) (spatial hinge), (T^*) (temporal hinge), and (τ_0) (proper-time unit) are constants declared at the hinge.

- These budgets are then checked against the discrete metric identity:

$$\begin{aligned} &[\\ &\Delta t_k^2 = \Delta \tau_k^2 + \frac{\Delta x_k^2}{c^2}, \\ &] \end{aligned}$$

with (c) a fixed unit-speed constant.

Crucially:

- Budgets are **not** used as free knobs to shape candidate selection; they are derived from the chosen candidate and then checked.

- Gravity (via ParentGate) influences **which candidates survive** and therefore indirectly affects the statistics of budgets, but it does not directly overwrite or bypass the budget relation.

6.5.2 Gravity Cannot Break the Invariant Interval

The engine enforces the discrete invariant interval for every accepted act:

$$\left[\Delta t_k^2 = \Delta \tau_k^2 + \frac{\{\Delta x_k\}^2}{c^2} \right]$$

This enforcement is **prior to** and **independent of** any specific interpretation of the budgets as physical time and space intervals.

In particular:

- ParentGate is not allowed to accept any candidate whose implied budgets would violate this identity.
- If a candidate's path through shells or radial movement would require a budget triple $(\Delta \tau_k, \Delta t_k, \Delta x_k)$ that fails the invariant relation, that candidate must be rejected at or before the budget-check step.

From this, two key consequences follow:

1. **No superluminal effective speeds from gravity:**
 - Combining the invariant interval with the No-Skip Rule ensures that:
$$\left| \frac{\Delta x_k}{\Delta t_k} \right| \leq c$$
for any accepted act.
 - Gravity cannot “bend” histories in a way that yields effective speeds greater than (c); it can only change which admissible (sub- or near-luminal) histories are feasible.
2. **Cone structure preserved:**
 - The causal cone structure induced by the budgets is preserved under gravity.
 - ParentGate can focus histories toward or away from certain regions (e.g., near a surface or horizon), but the allowed transitions remain inside the same discrete cone defined by the invariant relation.

In this sense, gravity is **implemented under** the SR-like interval structure, not in conflict with it.

6.5.3 Interaction Between ParentGate and Budgets

ParentGate influences budgets only indirectly, via which candidates survive:

- Candidates that attempt “radially expensive” transitions (e.g., large outward moves into stricter shells) are often rejected by ParentGate.
- As a result, **accepted** transitions tend to:
 - have smaller effective (Δx) for a given (Δt) ,
 - or, in some configurations, combine radial and tangential moves in ways that mimic curved paths around containers.

However, ParentGate does **not**:

- Introduce any additional terms into the budget identity.
- Modify ℓ_{H} , T^* , τ_0 , or (c) at runtime.
- Directly “add” a gravitational potential to (Δt) or $(\Delta \tau)$.

Instead:

- Gravity appears as a **statistical feature** of the accepted history ensemble:
 - certain paths through configuration space (e.g., those that remain near the container surface or follow certain shell-dependent patterns) are far more feasible than others;
 - over many steps, this biases worldlines in ways that, when interpreted in a coarse-grained picture, resemble trajectories in a curved spacetime or in a potential well.

6.5.4 No-Skip Rule and Light-Cone Structure Under Gravity

The engine’s **No-Skip Rule** (strict $(k \rightarrow k+1)$ locality) remains in force regardless of gravitational configuration:

- Each step is a small, local change; large “jumps” (either in tick index or in configuration space) are decomposed into sequences of local steps.
- Combined with the discrete invariant interval, this guarantees that:
 - **Effective light-cone structure** emerges even in the presence of gravity;
 - Gravity may tilt or focus the distribution of accepted paths but cannot create non-local, instantaneous influences.

Concretely:

- In simulations that include gravity via ParentGate, signals and influences still propagate through the network of present-acts at speeds $(\leq c)$.
- Gravity modifies which sequences of local moves are feasible but never allows causal shortcuts that would bypass intermediate sites or violate cone constraints.

6.5.5 Summary: Gravity Within the Budget and Constraint Framework

Putting these pieces together:

- Typed budgets ($(\Delta \tau, \Delta t, \Delta x)$) are derived from discrete counts and must satisfy the invariant interval for every accepted act.
- ParentGate applies shell-based feasibility filtering, modulated by χ , to decide which candidate transitions are allowed near containers.
- Gravity **influences** the statistics of budgets (which combinations occur frequently) but **cannot**:
 - violate the discrete metric identity,
 - induce superluminal effective propagation, or
 - circumvent the No-Skip Rule.

Therefore, in this framework, gravity is **fully subordinate** to the discrete SR-like structure defined by the budgets and engine constraints:

- SR-like behaviour (light-cone structure, maximum speed (c)) arises first from the invariant interval and No-Skip Rule.
- Gravity is then layered on as **feasibility modulation** consistent with that structure, leading to curved, deflected, or redshifted histories—without ever breaking the underlying relativistic budget constraints.

7. Quantum Structure and Measurement

7.1 Structural Quantum Mechanics in V1

In this framework, quantum behaviour is treated as a **structural feature** of the present-act dynamics, not as an extra postulate layered on top of a classical substrate. The core claim is that familiar quantum phenomena—superposition, interference, and Born-rule statistics—arise from how the present-moment hinge resolves relational ties between inner basins and outer candidates, under the constraints defined earlier (PMS, IN/ON, present plane, and structural Born rule).

Superposition as multi-basin support at the hinge

At the V1 level:

- The Inner Network (IN) is partitioned into **basins** ($\{R_i\}$) corresponding to distinct, context-resolved outcomes (e.g., different pointer positions, detection channels, or macro-configurations).
- A state is in **superposition** not because multiple worlds “exist” simultaneously, but because:
 - more than one basin (R_i) is admissible at the hinge, and
 - each such basin carries a non-zero present amplitude ($v_i \in \mathcal{P}$) on the Present Plane.

The superposed state at the hinge is thus encoded as a finite set of basins with associated present-plane amplitudes, ($\{(R_i, v_i)\}$), subject to normalization and structural constraints (Section 3.6).

Interference as amplitude structure on the Present Plane

Interference is handled entirely within the Present Plane:

- The complex structure (J) on (\mathcal{P}) (Section 3.6) allows amplitudes to carry phase-sensitive information.
- When the operator algebra (and, later, engine-level gates) propagate amplitudes along different relational paths, these amplitudes can:
 - **add** or **cancel** when they recombine at basins, depending on their relative phases,
 - produce constructive or destructive interference patterns in the effective weights assigned to basins.

No continuous spacetime waves are required at the fundamental level; interference is a property of how amplitudes over basins on (IN) evolve and recombine under the present-act dynamics.

Born rule as structural tie-resolution

The **structural Born rule** in V1 states that outcome weights must be proportional to the squared norms of present-plane amplitudes:

- Given normalized amplitudes (a_i) associated with basins (R_i), the long-run frequencies (or single-event probabilities, in a stochastic realization) are:

[
 $P(R_i) \propto |a_i|^2$.
]
- This is **not** introduced as a separate axiom; it follows from:
 - the requirement that probabilities track the measure of IN basins,
 - the way amplitudes on (\mathcal{P}) respond to coarse-graining and context changes, and
 - consistency constraints (e.g., additivity over mutually exclusive sub-basins).

At the engine level, this structural Born rule is realized concretely by the **PF/Born ties-only** mechanism (Section 4.5), which:

- constructs a primitive stochastic kernel on a finite tie set of candidates,
- takes its Perron–Frobenius eigenvector,
- and uses the squared entries of that eigenvector as Born-style weights.

Thus, the PF/Born step is the **operational counterpart** of the structural Born rule: it implements probabilistic tie-breaking in exactly those situations where the formal theory predicts a genuine quantum-like branching.

Collapse as hinge-level selection

In this picture, “collapse” is a natural part of present-act dynamics:

- A collapse event corresponds to the hinge selecting one basin (R_j) from a set of co-eligible basins ($\{R_i\}$).
- The selection is:
 - fully deterministic when the structural ordering (gates, residuals, fewest-acts) yields a unique winner,
 - stochastic—governed by the PF/Born ties-only rule—only when multiple basins remain **structurally identical** after all deterministic filters.

Once a basin (R_j) is selected:

- Its content is sunk into (IN) as new record,
- The amplitude structure is reset to reflect that only (R_j) is now actual at that hinge,
- The Outer Network (ON) is renewed accordingly, generating fresh potential for the next present-act.

There is no separate “collapse postulate”; this selection is simply how a single present-act resolves what was previously encoded as a multi-basin, amplitude-bearing structure at the hinge.

Relation to standard quantum mechanics

From the V1 standpoint, standard quantum mechanics appears as:

- A **continuum approximation** in which:
 - the Present Plane is extended to a full Hilbert space,
 - amplitudes are allowed to live in high-dimensional spaces,
 - and dynamics are idealized as continuous unitary evolution followed by Born-rule measurement.

The present-act framework instead provides:

- a **discrete, present-centred** version of this story, in which:
 - amplitudes live on a minimal present-plane structure tied to IN basins,
 - evolution is driven by a finite alphabet of operators acting on Tick-State Carriers,
 - and Born-rule statistics arise from tie-resolution at the hinge rather than from an external measurement postulate.

Subsequent subsections describe how this structural quantum picture is realized in the engine (V2) and how measurement and framing are handled via Collective Spheres and frames.

7.2 Engine-Side Quantum Behaviour

At the engine level, the quantum structure from V1 is realized directly in the way the present-act engine selects outcomes. The same finite, discrete machinery that governs all acts—selectors, gates, budgets, PF/Born ties-only—is sufficient to reproduce key quantum-like behaviours when configured appropriately.

Superposition and interference in the engine

In V2:

- A **superposed** situation corresponds to a step where the candidate set ($\mathcal{C}_k^{\{\text{feas}\}}$) contains multiple hinge-admissible, structurally indistinguishable candidates that:
 - survive all non-gravity gates,
 - survive ParentGate,
 - share the same residual triple and fewest-acts value,
 - and differ only in phase-like or relational structure encoded in (Ξ).
- In such cases, these candidates form the tie set (\mathcal{T}) passed into the PF/Born ties-only procedure. The adjacency rules and kernel (M) are chosen so that:
 - “in-phase” candidates reinforce each other,
 - “out-of-phase” candidates interfere destructively,
 - and the Perron–Frobenius eigenvector (v) encodes the resulting interference pattern.

When squared to obtain Born weights, ($|v_i|^2$) plays exactly the role of ($|a_i|^2$) in the structural Born rule. Thus, **interference fringes** in simple scenes (e.g., two-path or multi-path arrangements) appear as spatial or contextual variations in the frequencies with which particular candidates win the PF/Born tie.

Decoherence and effective classicality

Decoherence appears when:

- Additional structure (environmental degrees of freedom, context tags, CRA-like constraints) is included in (W_k) and (Q_k), and
- The gates and selectors are tuned so that different macro-contexts or environment states rapidly become **distinguishable** in (Ξ).

In that regime:

- Many candidate sets that would otherwise have produced a large tie set (\mathcal{T}) are **split** by structural or CRA-like gates into distinct context lanes.
- Within each lane, the tie sets shrink or disappear; the ratio-lexicographic and fewest-acts ordering alone often produces a unique winner.
- As a result, the PF/Born procedure is invoked rarely or only on small local ties, and macro outcomes appear **effectively deterministic** and classical.

In other words, decoherence is not a separate mechanism; it is the natural result of:

- increasing the environment’s role in the state representation, and
- enforcing stricter context-resolved admissibility and structural conditions via the existing gates.

No-signalling and locality

The engine is explicitly built to satisfy a **no-signalling** condition at the control level:

- All candidate construction, gating, and ordering decisions at a site (k) depend only on:
 - local (W_k) and (Q_k),
 - their discrete tags in ($\backslash X_i$),
 - local context-level and frame data from the manifest.
- There is no gate or selector that:
 - directly inspects remote states outside the local neighbourhood, or
 - conditions a local decision on remote context choices in a way that would permit superluminal signalling.

In particular:

- The PF/Born ties-only step:
 - operates only on the **local tie set** ($\backslash \mathcal{T}$),
 - uses an adjacency kernel built from local, discrete structure,
 - and uses a local RNG seed determined by the current site and manifest.
- Changing settings or configurations in a space-like separated region alters the local candidate sets there, but does not allow an agent to steer the marginal outcome distribution at a distant site in a controllable way.

This reproduces the **no-signalling** pattern seen in quantum theory: correlations can be non-classical, but they cannot be used to send controllable signals faster than (c).

Complementarity and context choice

Complementarity between “which-path” and “interference” descriptions is implemented via the **context and structural gates**:

- When the manifest enables a **which-path tag** in ($\backslash X_i$) and CRA-like gates enforce that this tag is preserved, the engine:
 - distinguishes between different paths at the structural level,
 - prevents those paths from forming a single tie set ($\backslash \mathcal{T}$) at the hinge,
 - and thereby suppresses interference between them (no significant tie set, little or no PF/Born invocation).
- When the manifest instead:
 - avoids encoding which-path information, or
 - allows gates to erase it before the hinge is evaluated,

the engine can produce larger tie sets where different path candidates interfere via the PF/Born mechanism, generating interference fringes.

Thus, **complementarity** is realized as a trade-off between:

- making certain distinctions explicit in ($\backslash X_i$) and in context tags (leading to path-distinguishability and reduced interference), and
- allowing ties to remain at the structural level (leading to interference patterns in PF/Born outcomes).

Summary

On the engine side:

- The same discrete, finite machinery used throughout the present-act engine is sufficient to reproduce:
 - superposition (via multi-basin support and tie sets),
 - interference (via adjacency kernels and PF eigenvectors),
 - decoherence / effective classicality (via environment and CRA-like gates), and
 - no-signalling and complementarity (via locality of control and context-resolved admissibility).

No separate “quantum add-on” is required; quantum behaviour is embedded in how the engine resolves present-acts at the hinge under the constraints established by the ontology, formal core, and context ladder.

7.3 Measurement and Framing in Collective Spheres

In this framework, **measurement** is not a separate physical process added on top of the dynamics; it is a special case of **framing** and **context selection** carried out within Collective Spheres (CS), using the same present-act and gate machinery described earlier.

7.3.1 Measurement as Framing in a Collective Sphere (CS)

A **measurement setup** is represented as:

- A **Collective Sphere (CS)** that includes:
 - the system being “measured”,
 - the measurement apparatus (pointer, detector, etc.),
 - any relevant environment structure, and
 - one or more observer streams whose PMSs are synchronized to that CS.
- A **frame** attached to that CS that specifies:
 - which coarse-grained variables count as **pointer states**,
 - which bands and shells are relevant for the measurement,
 - which present-plane basins ($\{R_i\}$) correspond to distinct outcome channels.

Formally, a measurement is then:

A framing operation (via CT and C-like sync operators) that establishes a CS with a particular outcome partition ($\{R_i\}$) and then lets the ordinary present-act dynamics select one basin and sink it into (IN_{CS}) as shared record.

No extra “measurement postulate” is introduced; measurement is a **CS-framed present-act**.

7.3.2 Measurement-Induced Context Collapse (MICC-style gates)

At the engine level, **MICC-style gates** implement **Measurement-Induced Context Collapse**:

- They act as specialized structural/CRA-like gates that:
 - enforce that each candidate corresponds to one of a discrete set of **pointer-compatible contexts** (e.g., pointer left, pointer right, no-click, etc.),
 - prevent candidates from mixing contexts that the measurement frame intends to keep distinct,
 - and ensure that, after collapse, the CS-level record (IN_{CS}) reflects a single, well-defined outcome channel.

Operationally:

- Before the PF/Born ties-only step, MICC-style gates:
 - partition the candidate set into lanes corresponding to distinct outcome contexts,
 - discard candidates that are structurally inconsistent with any allowed lane,
 - thereby reduce the tie set (\mathcal{T}) to those candidates that differ only within a given outcome lane.

The result is:

- A **context-resolved collapse**: once a basin (R_j) (and its associated lane) is selected, the CS record treats that lane as “the outcome”, and other lanes are not present in (IN_{CS}) for that act.

7.3.3 Pointer States and Classicalization

Pointer states are coarse-grained configurations in the CS that behave as stable, classical-looking records:

- In the formal core, they appear as basins (R_i) on (IN_{CS}) that:
 - are robust under small perturbations (multiple micro-configurations map to the same basin),
 - have well-defined coarse features in (Ξ) (e.g., pointer position, detector flag),
 - and are embedded in frames with strong synchronization conditions (CS-wide agreement).
- In the engine, pointer states correspond to:
 - specific patterns in (W_k) and (Q_k) that are tagged as “pointer-like” in (Ξ),
 - structural gates that heavily penalize deviations from these patterns once a basin is selected,
 - context-resolved admissibility rules that prevent mixing of distinct pointer basins in subsequent acts.

Classicalization is then:

The process by which repeated present-acts, under MICC-style gating and environmental coupling, drive the system into basins where pointer configurations are stable under the engine's gates and context rules.

In practice:

- After a measurement-like event, subsequent acts see:
 - a sharply reduced tie set (often a single pointer basin),
 - strong structural and context constraints that preserve this basin,
 - and minimal further invocation of PF/Born in that channel.

Macroscopically, this produces the familiar behaviour:

- Pointers and macroscopic records appear to follow **deterministic**, classical dynamics after an outcome is registered, even though the underlying framework still supports quantum-like interference and stochasticity in situations where multiple basins remain structurally tied.

7.3.4 Role of Frames and CS Synchronization in Multi-Observer Measurements

When multiple observers are involved:

- Their PMSs are members of the same CS, synchronized to a shared frame.
- Measurement framing (CT + Sync) ensures that:
 - the same basin (R_j) is recorded in (IN_{CS}),
 - each observer's local (IN_k) is updated to be consistent with that CS-level record,
 - subsequent acts for each stream treat that basin as part of their local record.

This synchronization mechanism:

- Realizes **agreement on outcomes** as a purely structural effect:
 - no separate “wavefunction collapse” needs to be postulated;
 - consistency across observers follows from their shared CS and frame constraints.

In summary:

- Measurement is implemented as **framing plus MICC-style gating** in a CS.
- Pointer states and classicalization emerge from how basins are selected and stabilized under these gates.
- Multi-observer agreement is enforced by CS synchronization, not by an extra global collapse rule.

8. Simulations and Evidence Programme

8.1 Simulation Philosophy and Reproducibility

The simulation and evidence programme is designed to test the framework at three levels:

- **Internal structural consistency** of the formal core (V1) and engine (V2).
- **Qualitative and semi-quantitative behaviour** (e.g., SR-like kinematics, interference, decoherence, feasibility-based gravity) in controlled scenes.
- **Comparison to external data** (e.g., rotation curves, radial acceleration relations, lensing) where the framework makes specific, testable predictions.

The goal is not to exhaustively fit all available data, but to demonstrate that:

- The present-act engine can reproduce the **main qualitative patterns** expected from special relativity, quantum mechanics, and weak-field gravity under a single, unified set of structural rules; and
- Key constructions such as the context ladder, UGM, and χ are consistent with, and in some cases constrained by, observed phenomena.

Simulation layers

The programme is organized into three main layers:

- **V1 simulations**
 - Use abstract V1 objects (carriers, PMS/IN/ON/CS, operator algebra, pivot profile) to test:
 - operator identities and algebraic properties,
 - behaviour of the context ladder and pivot function ($g(D)$),
 - simple “field-like” modules (e.g., gauge-like behaviour, curvature translators).
- **V2 simulations**
 - Use the concrete present-act engine with finite (W_k , Q_k , $\backslash X_i$), gates, budgets, ParentGate, and PF/Born ties-only to test:
 - engine hygiene (locality, no-skip, SR-like budgets, no-signalling),
 - decoherence and classicalization in measurement-like setups,
 - gravity-as-feasibility scenes (plateaus, horizons, deflection, delay, redshift),
 - matter-addition suites and EM/gravity analogues in context-level scenes.
- **External-data tests (T-series)**
 - Apply the engine and CL structure to real or realistic data, including:
 - rotation curves and radial acceleration relations (RAR),
 - galaxy–galaxy lensing and Milky Way–scale activation at the $+2 \leftrightarrow +3$ seam,
 - cross-scale alignment of hinge scales (UGM, CNS size band, perceptual resolution).

Each layer is documented in the attached simulation and evidence volumes (V1 Simulations, V2 Simulations, Core Evidence Narrative, and CL probe reports), which are incorporated by reference into this defensive publication.

Reproducibility and archival code

All simulation results referenced in this framework are intended to be **reproducible** by a person skilled in the art, using the materials in the v0.9 bundle:

- The archived bundle includes:
 - engine source code and core libraries,
 - manifests and configuration files used for each simulation family,
 - driver scripts or notebooks that set up and run the V1 and V2 simulations,
 - selected output data and plots for verification.
- For each major simulation family, the attached V1 and V2 simulation documents:
 - describe the purpose of the family (what it is testing),
 - list the relevant configuration and parameter choices,
 - specify which code modules and manifests are used,
 - and summarize the outcomes (pass, obstructed, pending) and their interpretation.

To reproduce a given simulation family, a verifier can:

1. Locate the corresponding entry in the V1 or V2 simulation document.
2. Use the file and module names given there to locate the code and configuration within the v0.9 archive.
3. Run the specified scripts or notebooks with the declared parameters.
4. Compare the resulting diagnostics (e.g., plots, tables, acceptance ratios) with those recorded in the documentation.

The **cryptographic hash** of the v0.9 archive (Section 0.3) ensures that the code and configuration files used for these simulations are fixed and verifiable. No simulations or results that rely on code outside this hashed archive are treated as canonical for the purposes of this defensive publication.

Scope and limitations

The simulation and evidence programme as recorded in v0.9:

- Provides a **broad set of structural and phenomenological tests**, including:
 - operator-core checks,
 - ladder and pivot behaviour,
 - SR-like budget behaviour,
 - quantum-like interference/no-signalling,
 - matter-addition and gravity/feasibility scenes,
 - and first-round external-data applications (rotation curves, RAR, lensing).
- Does **not** claim:

- to fully exhaust all possible simulation regimes,
- to provide final, best-fit parameter sets for all observational data, or
- to cover every edge case implied by the theory.

Where simulations are marked as **obstructed** or **pending**, this is explicitly stated in the attached simulation documents, along with notes on what additional work would be required to complete those lines of investigation.

8.2 V1 Simulation Families (Structural Tests)

The V1 simulation programme tests the **formal core** of the theory—PMS/IN/ON/CS structures, operator algebra, context ladder, and pivot behaviour—without yet invoking the full V2 engine. These simulations are organised into families, each targeting a specific structural question.

8.2.1 Operator-Core and Algebra Tests

These simulations check that the primitive operators and their compositions behave as required:

- **Tick and operator algebra checks**
 - Verify that compositions of Renew (F), Sink (S), Trade (T), Sync (C), and Framing (CT) respect:
 - monotonic growth of IN,
 - conservation of capacity K (ledger identity),
 - non-commutation properties (e.g., $F \circ S \neq S \circ F$ in general),
 - and closure under admissible compositions.
- **Arrow-of-time and ledger evolution**
 - Confirm that along admissible paths:
 - record budget (I_k) is non-decreasing,
 - exposure budget (E_k) adjusts in accordance with operator roles,
 - no composition produces a net decrease in (I_k) or a violation of ($I_k + E_k = K_k$).

These tests are mostly algebraic and combinatorial; they ensure that the V1 operator definitions are internally consistent and match the intended semantics.

8.2.2 Ladder and Pivot Behaviour

This family tests the **context ladder** and **pivot function** ($g(D)$) in simplified settings:

- **Dimension profile sweeps**
 - Simulations where the inner dimension ($D(n)$) is swept across a range of values for each context level (n), and the resulting pivot values ($g(D(n))$) are analysed.
 - Check that hinge-like behaviour appears near the designated hinge dimension ($D_{\text{ast}} \approx 2$) and that inner/outer levels exhibit distinctly non-hinge-like behaviour.

- **Pivot profile over n**
 - Construct numerical examples of $(D(n))$ and $(g(D(n)))$ across a small ladder (e.g., $-3 \dots +3$) and verify:
 - hinge-like values near $(n = 0)$,
 - appropriate asymmetries between inner machinery (negative n) and outer containers (positive n),
 - and robustness of the hinge profile under modest perturbations.

These tests validate that the ladder and pivot definitions produce the intended hinge and container roles at the formal level.

8.2.3 Gauge-Like and Field-Like Modules

Several V1 simulations implement simplified **field-like** modules to test whether:

- A PMS/IN/ON/CS structure with a tick algebra can:
 - support gauge-like behaviour,
 - represent “field lines” and potentials as structural patterns in ON/CS,
 - and maintain invariances analogous to gauge symmetries under certain operator sequences.

Typical tests include:

- **Toy lattice-gauge modules**
 - Present-moment configurations arranged on small discrete lattices, with operator sequences that update link variables subject to constraints.
 - Check that certain combinations of updates correspond to “pure gauge” transformations (no observable change in IN), while others produce physically distinct patterns.
- **Compact curvature translators (prototype gravity modules)**
 - Simple constructions that attempt to translate kernel-like structures on IN/ON into effective “curvature” signals, serving as early tests of how gravity-like behaviour might appear in the present-act framework.

Results from these modules inform how later, more complete engine-level implementations (in V2) handle EM-analogue and gravity-analogue behaviour.

8.2.4 Early Measurement and Classicalization Experiments

These simulations explore **measurement-like** scenarios directly in V1 terms:

- **Two-basin and multi-basin setups**
 - Construct situations where IN has two or more basins corresponding to distinct outcomes, and amplitudes on the Present Plane are assigned accordingly.
 - Apply V1 operator sequences that mimic “measurement” actions (framing, sync, and sink operations) and check whether:

- a single basin is selected and sunk into IN,
- the resulting record behaves as expected under subsequent operators.
- **Classicalization tendencies**
 - Test whether repeated application of certain operator patterns can stabilize particular basins, making them effectively classical pointer states at the formal level.

These experiments serve as proofs-of-concept for the structural measurement picture later realized explicitly in V2.

8.2.5 Status and Use of V1 Simulation Results

In the v0.9 bundle, V1 simulations are used to:

- Validate that the **formal definitions** (operators, ladder, pivot, present plane) are internally consistent and behave qualitatively as intended.
- Identify which constructions are robust and which require refinement (e.g., early compact-curvature translators that did not produce satisfactory results are recorded as **obstructed** or **superseded** by later V2-based approaches).
- Provide a **baseline** against which V2 engine simulations can be compared: V2 is expected to reproduce, in discrete engine form, the core structural behaviours that V1 demonstrates in abstract.

The V1 simulation families are documented in the attached “V1 Simulations” volume, with code and configuration references in the v0.9 archive. They form the first layer of evidence that the present-act framework can be consistently realized and manipulated as a formal system before moving to full engine implementations and external-data tests.

8.3 V2 Engine Simulation Families

The V2 simulation suite tests the **fully instantiated present-act engine** (discrete sites, (W_k, Q_k, χ_i) , gates, budgets, ParentGate, PF/Born ties-only) in progressively more demanding settings: internal quantum/SR integrity, emergent field dynamics via matter-addition, hinge-scale gravity with a single χ , and external-data tests.

8.3.1 Core Internal Simulations: Quantum & SR Integrity (Q-Series)

These simulations verify that the engine, under its discrete rules, reproduces key quantum-like and relativistic behaviours without metric fields, potentials, or continuous weight functions in the control path.

- **Q1 – Gravitational Interferometer**
 - Constructs a two-path or multi-path interferometer scene in which feasibility gradients (via ParentGate) play the role of “gravitational potential” variations.

- Demonstrates interference fringes in the distribution of PF/Born-selected outcomes, entirely from discrete feasibility and adjacency rules; no wavefunctions or metric tensors are used in the engine's control logic.
- **Q2 – Focusing (Fermat-type geodesics)**
 - Uses feasibility gradients defined by ParentGate to mimic refractive index or potential variations.
 - Shows that the most feasible paths concentrate near those that would be geodesics or least-time paths in a continuum model (Fermat-type behaviour), again without solving differential equations.
- **Q3 – Horizons and Trapping Regions**
 - Introduces sharp increases in strictness schedules (via ParentGate) at certain shell indices to create horizon-like boundaries.
 - Finds that beyond such thresholds, histories attempting to escape are heavily suppressed: the interior becomes effectively neutral with respect to escape, reproducing horizon-style behaviour in purely discrete terms.
- **Q4 – Local CHSH / Bell-Type Tests Under Gradients**
 - Configures paired subsystems under feasibility gradients and tests local CHSH-type inequalities.
 - Demonstrates that the engine can reproduce Bell inequality violation patterns while maintaining strict locality and no-signalling, even in the presence of feasibility gradients (no superluminal signalling channels are introduced).

In all Q-series tests, SR-like budget identities and no-skip locality remain strictly enforced. Quantum-like effects arise from the PF/Born ties-only rule and discrete adjacency structure, not from continuous fields.

8.3.2 Interference and Decoherence

Separate simulations focus on **interference vs which-path information** and on **decoherence**:

- **Complementarity tests**
 - Scenes are run both with and without explicit which-path tags in ($\backslash X_i$) and corresponding CRA-style gates.
 - When which-path information is encoded and protected, tie sets shrink and interference is suppressed; when it is erased or not encoded, tie sets grow and interference fringes re-emerge in PF/Born outcomes.
- **Decoherence tests**
 - Enlarged environment records (extra context tags and environment pixels in (W_k, Q_k)) are introduced.
 - Boolean “which-environment” marks and CRA gates quickly split candidate sets into disjoint lanes, reducing the occasions where PF/Born is invoked and producing effective classical behaviour for macroscopic variables.

These internal tests confirm that the engine reproduces the standard complementarity and decoherence patterns using only combinatorial gate logic and PF/Born ties-only, consistent with the structural quantum description in Section 7.

8.3.3 Emergent Field Dynamics via Matter-Addition Suite (D/E/F-Series)

The **matter-addition suite** tests whether field-like EM and gravity-like behaviours emerge from **counts-only diagnostics** when “matter” is introduced as a pattern of strictness and feasibility—not as a separate force field.

- **Method of Matter-Addition**
 - “Matter” is introduced by modifying local gate conditions and ParentGate strictness around chosen regions, not by inserting explicit field variables or source terms into the control logic.
 - Diagnostics read off **counts of accepted worldlines** or configurations as a function of position, shell index, and time; no differential equations are solved in the engine core.
- **Electromagnetic Analogues (D-Series)**
 - **Gauss plateau:** counts-only diagnostics around a localized “charge-like” configuration exhibit a plateau region and a clear $(1/r)$ -type falloff in counts, mimicking Gauss-law behaviour.
 - **B-circulation:** magnetic-like circulation patterns are reproduced via sequences of discrete “re-expression” (R3) events, rather than explicit magnetic field vectors; circulation emerges in counts around loops.
 - **Polarization:** detector sectors track E-like and B-like count patterns with phase lags and polarization-angle dependence in a ring of propagation sites, again read off from counts, not from continuous field variables.
- **Gravity-Style Geometry (E-Series & F-Series)**
 - **Redshift:** frequency ratios of signals passing through shell-dependent feasibility gradients show gravitational-redshift-like scaling, with ratios matching those predicted from the feasibility profile.
 - **Shapiro delay:** traversal times measured against impact parameters (b) exhibit a Shapiro-like logarithmic delay ($\Delta t \propto \log b$), produced purely by shell-based feasibility constraints.
 - **Deflection:** ray paths passing near a central strictness region show weak-deflection angles scaling approximately as $(1/b)$, consistent with weak-lensing behaviour.
 - **Mesh Certification & Re-centering:** geometric results are verified by mesh-based audits and re-centering tests to ensure invariance under changes of coordinate origin and discretization choices.

These families demonstrate that EM-like and gravity-like “fields” emerge as **counts-only summaries** of the engine’s feasibility-filtered discrete dynamics.

8.3.4 Hinge-Scale Gravity Triad and T-Series External-Data Tests

The **hinge-scale gravity triad** (V2.1) and associated **T-series** simulations test whether a **single amplitude family** χ , derived from context scales, can simultaneously account for several gravitational observables.

- **Hinge-Scale Gravity Triad (V2.1)**
 - Uses the χ defined from $((L_{\text{UGM}}, R_{\text{oplus}}, R_{\text{obs}}))$ to set the overall strictness schedule for ParentGate around Earth-like containers.
 - Tests three key weak-field observables (the “triad”) in engine scenes:
 - **Deflection:** weak-deflection angles for rays passing near a container,
 - **Time delay:** Shapiro-style delays as a function of impact parameter,
 - **Redshift:** gravitational redshift-like frequency changes across shells.
 - A single χ -family is used; there is **no per-observable re-tuning**. All three behaviours must fall into the pre-calibrated amplitude band at once.
- **T-Series External-Data Tests (Rotation, RAR, Lensing, Activation)**
 - **T1/T2 – Rotation Curves & Radial Acceleration Relation (RAR):**
 - Apply the engine’s scaling and CL principles to galaxy rotation data.
 - Use flat-window criteria and hinge-scale arguments to automatically identify plateau regions without per-galaxy tuning.
 - Recover an emergent RAR slope in the low-acceleration regime $(g_{\text{obs}} \propto \sqrt{g_{\text{bar}}})$, consistent with observed RAR behaviour, using a single global acceleration scale.
 - **T3 – Galaxy–Galaxy Lensing Plateaus:**
 - Use lensing profiles binned by stellar mass and size.
 - Identify plateau-like regions where lensing amplitude flattens in a way consistent with shell-based feasibility.
 - Observe that mid-range mass bins show increased lensing amplitude despite lower average densities, consistent with a size-mediated activation of feasibility gradients.
 - **T3-B – Milky Way Activation Model:**
 - Test a “Size + Activation” model where lensing effectiveness jumps when galaxies exceed a Milky Way–like size threshold.
 - Compare a Size + Activation model to Size-only models using standard model-selection tools (e.g., AIC).
 - On the DR5 dataset, find evidence for a preferred activation scale $(R_{\text{MW}} \approx 6)$ kpc and an associated increase in lensing amplitude beyond this threshold, consistent with a $+2 \leftrightarrow +3$ context activation at Milky Way scale.

Taken together, these results show that:

- The **same χ -family** that controls weak-field behaviour in idealized engine scenes also provides a reasonable match to rotation and lensing data,
- Without introducing separate phenomenological parameters for each effect or each galaxy.

8.3.5 Status and Role of V2 Simulations

In the v0.9 bundle, V2 simulations:

- Demonstrate that the **present-act engine** can reproduce:
 - SR-like behaviour (invariant interval, speed limit, no-skip locality),
 - quantum-like interference, complementarity, and decoherence,
 - EM-like and gravity-like field behaviours from counts-only diagnostics,
 - and a first round of astrophysical observables (rotation curves, RAR, lensing plateaus, Milky Way-scale activation),

using a single coherent engine specification and a structurally derived χ .

- Mark clearly:
 - Which families are **well-behaved and robust**,
 - Which are **sensitive** to mesh, manifest, or parameter choices,
 - Which remain **obstructed or incomplete**, along with notes on what further work is required.

All of these simulations are documented in the attached “V2 Simulations” and “Core Evidence Narrative” volumes, with code and manifest references in the hashed v0.9 archive. They form the central empirical and numerical support for the claim that gravity, quantum behaviour, and relativistic structure can all be expressed within a single present-act engine under the constraints laid out in this defensive publication.

8.4 Limitations and Open Simulation Directions

The simulation and evidence suite in the v0.9 bundle is **deliberately first-pass**. It is extensive enough to test the main structural claims of the framework, but it is not a complete exploration of all regimes implied by the theory. This subsection records the main limitations and the most important open directions.

8.4.1 Internal Engine and Structural Limitations

- **Finite scene size and resolution**
 - All V2 simulations are run on finite grids or graph structures with manifest-declared limits on resolution, shell counts, and context bands.
 - No simulation in v0.9 attempts a full global model (e.g., complete galaxy population or full cosmological volume); instead, each scene is a local or meso-scale test of a specific mechanism (interference, plateaus, horizons, etc.).
- **Approximate container models**
 - Containers (e.g., Earth, galaxies, cosmic shells) are represented by simplified shell structures and strictness schedules, not by detailed density profiles.

- Earth is typically modelled as an approximately spherical container; galaxies are often modelled as axisymmetric disks; cosmic shells as idealized outer boundaries.
 - The current strictness schedules are minimal patterns designed to test feasibility-geometry ideas, not fully realistic mass/structure distributions.
- **Approximate hinge and ladder calibration**
 - The UGM and temporal hinge scales are treated as **hinge bands**, not single exact values, and are implemented via discrete binning choices that approximate those bands.
 - Context bands are specified using relatively coarse ranges; finer calibration of band boundaries and transitions is left as future work.
- **Early or superseded V1 modules**
 - Some V1-era curvature-translator and field-module simulations are recorded as **obstructed** or superseded by V2-based implementations (e.g., compact-curvature translators that did not produce robust or interpretable results).
 - These are kept in the record but are not treated as confirmed successes; instead, they motivate the V2 matter-addition and gravity-feasibility suite.

8.4.2 External Data and Fitting Limitations

- **Limited datasets and proxies**
 - The T-series tests (rotation curves, RAR, lensing) are run on specific public datasets (e.g., subsets of rotation curves and lensing catalogs) and, in some cases, on derived summary quantities (e.g., binned lensing amplitudes, accelerations).
 - No attempt is made in v0.9 to process all available data sources or to fit every individual galaxy or lensing system.
- **Single-parameter χ family**
 - The gravitational amplitude χ is treated as a **single amplitude family** derived from ladder scales, not as a free multi-parameter fit.
 - Comparisons to GR amplitudes and to data (deflection, delay, redshift, rotation, lensing) are therefore necessarily approximate; the goal is to test whether the structurally derived χ family falls in the right range, not to perform detailed, per-system parameter fitting.
- **Error modelling and uncertainties**
 - The v0.9 simulations and evidence docs include basic uncertainty estimates (e.g., error bars from data or from sampling variability), but they do not yet implement full Bayesian or frequentist error-propagation frameworks for all observables.
 - Some comparisons are therefore presented as qualitative or semi-quantitative “ballpark” checks rather than as rigorous statistical tests.

8.4.3 Incomplete or Pending Simulation Lines

Several simulation lines are explicitly marked as **pending** or **incomplete** in the v0.9 documentation, including but not limited to:

- **Extended matter-addition suites**

- Further exploration of EM-analogue and gravity-analogue behaviour in more complex scenes (e.g., multiple interacting sources, time-dependent sources, nontrivial topologies) is planned but not yet fully carried out.
- **Refined T-series tests**
 - Follow-up tests on:
 - more detailed rotation-curve datasets,
 - alternative RAR samples,
 - and expanded lensing catalogs,
 are outlined but not yet fully implemented in v0.9.
- **Full pointer-dynamics and measurement suites**
 - While basic measurement and classicalization scenes are tested, a full catalogue of pointer-dynamics simulations (covering a wide variety of measurement setups and environment couplings) remains an open task.

These lines are documented in the attached simulation volumes with notes on what additional code, data, or analysis steps would be required to complete them.

8.4.4 Planned Extensions and Refinements

The v0.9 bundle also records several **planned extensions** that are explicitly left as future work:

- **Improved ladder and hinge calibration**
 - Refining the mapping between abstract context levels and physical scales, including:
 - more accurate CL mapping for non-terrestrial systems,
 - improved calibration of UGM and temporal hinge against biological and psychophysical data,
 - and clearer constraints on band boundaries from cross-scale probes.
- **Refined χ construction and container modelling**
 - Exploring:
 - alternative but structurally consistent definitions of χ that incorporate more detailed pivot information or additional context scales,
 - more realistic container models (e.g., non-spherical mass distributions, multi-component galaxies), and
 - the impact of these refinements on deflection, delay, redshift, rotation, and lensing predictions.
- **Larger-scale engine runs**
 - Extending current scenes to:
 - larger grids/graphs,
 - multiple interacting containers,
 - longer run times to probe emergent behaviour in more complex environments.
- **Formal verification and automated auditing**
 - Strengthening and automating:
 - curve-ban checks,
 - diagnostics-leak audits,

- PF/Born integrity checks,
- locality and no-skip verification,
- and invariant-interval consistency tests.

8.4.5 Role of v0.9 Simulations in This Defensive Publication

Within this defensive publication, the simulations recorded in the v0.9 bundle should be understood as:

- **Evidence of feasibility and structural coherence** – they show that the present-act engine, under the stated constraints, can reproduce a wide range of expected behaviours and make contact with real data without ad hoc forces or potentials in the control path.
- **Not yet a complete empirical programme** – they do not exhaust all observational tests or all parameter regimes implied by the theory, nor do they claim final quantitative precision.
- **A foundation for extension** – they define a clear set of next steps (as outlined above) that can be pursued to sharpen, confirm, or refute specific aspects of the framework.

All of these limitations and open directions are recorded explicitly in the attached V1 simulations, V2 simulations, and Core Evidence Narrative volumes, which are incorporated here by reference.

9. Implementation and Reproducibility

9.1 Code Archive Overview

The canonical implementation of the present-act engine and its associated tooling at the v0.9 snapshot is contained in the cryptographically hashed archive:

- **Archive name:** A.R. V2 - v0.9.zip

This archive is the **only** implementation bundle referenced by this defensive publication. It is the source of truth for:

- The engine's code and data structures,
- The manifests and configuration files used in the simulations,
- The scripts and notebooks used to run the V1 and V2 simulation families,
- The diagnostic and audit tooling used to verify engine constraints.

The archive is fixed by the SHA-256 hash given in Section 0.3. Any copy with that hash is, by definition, an exact copy of the implementation and simulation environment referred to here.

9.1.1 Contents of the Archive (High-Level)

At a high level, the archive is organised into:

- **Engine Core**
 - Source code implementing the V2 present-act engine:
 - Site representation and state objects ($(W_k, Q_k, \text{budgets}, \text{tags})$),
 - Feature alphabet (\mathcal{X}) and feature-map machinery,
 - Selector implementations,
 - Non-gravity gates (Θ, κ , structural, CRA-like),
 - ParentGate (gravity feasibility gate) and shell structures,
 - Ratio-lexicographic ordering and fewest-acts tiebreak,
 - PF/Born ties-only module and RNG integration,
 - Budget computation and invariant-interval enforcement.
 - Any shared libraries and utilities used by the engine (e.g., adjacency-graph builders, kernel constructors, logging utilities).
- **Manifests and Configuration Files**
 - One or more manifest files that declare:
 - The feature alphabet (\mathcal{X}) and tag schema,
 - Context-level band boundaries and hinge scales,
 - Gate definitions, thresholds, and shell schedules,
 - Values of key structural constants (e.g., c), hinge-related unit scales).
 - Configuration files for specific simulation families (e.g., scene geometry, initial conditions, number of shells, number of sites, runtime parameters).
- **Simulation Drivers and Notebooks**
 - Scripts or notebooks for:
 - V1 simulation families (operator-core, ladder/pivot tests, field-like modules, early measurement cases),
 - V2 simulation families (Q-series, interference/decoherence tests, matter-addition suites, T-series external-data tests).
 - Each driver defines:
 - Which engine modules to import,
 - Which manifest and configuration files to use,
 - How many runs/samples to generate,
 - Where to store output (e.g., logs, tables, plots).
- **Diagnostics and Audit Tools**
 - Code implementing:
 - Curve-ban audits (checking that control path uses only discrete predicates and ratios),
 - Diagnostics-leak audits (ensuring diagnostics do not feed back into control),
 - PF/Born integrity checks (ensuring randomness is ties-only),
 - Locality and no-skip checks (ensuring only local transitions and $(k \text{ to } k+1)$ steps),
 - Invariant-interval consistency tests on typed budgets.

- Scripts for running these audits on engine logs and manifests.
- **Selected Outputs and Reference Results**
 - Representative output files and plots from key simulation families, used as reference points for reproducibility:
 - Internal engine tests (e.g., no-signalling, SR-like behaviour),
 - Interference and decoherence scenes,
 - Matter-addition and EM/gravity-analogue tests,
 - T-series external-data fits (rotation curves, RAR, lensing, Milky Way activation).

9.1.2 Role of the Archive in This Defensive Publication

Within this defensive publication, the archive serves three functions:

- **Implementation Reference**
 - It is the canonical implementation of the present-act engine and associated tools at v0.9.
 - Any future implementation claiming to instantiate this framework can be compared against this code to check whether it respects the same structural constraints and algorithmic details.
- **Reproducibility Resource**
 - It enables a skilled person to reconstruct:
 - the engine behaviour described in Sections 4–7,
 - the simulations and evidence described in Section 8.
 - By using the drivers and manifests in the archive, one can rerun the simulation families and compare results to the documented outputs.
- **Integrity Anchor**
 - Its SHA-256 hash is recorded on-chain.
 - This ensures that the exact code and configuration used for the v0.9 simulations can be uniquely identified and verified in the future, even if later versions of the theory or engine are developed.

Subsequent subsections specify how to reconstruct the engine from the archive (Section 9.2), how manifests and parameter tables are structured (Section 9.3), and how diagnostics and verification tools are used to confirm that an implementation satisfies the finiteness, locality, curve-ban, and PF/Born integrity constraints (Section 4.6 and Section 9.4).

9.2 Engine Reconstruction Procedure (Step-By-Step)

This subsection specifies, at a high level, how a person skilled in the art can reconstruct a functionally equivalent present-act engine from the v0.9 archive and the definitions in this defensive publication. It assumes access to the archive `A.R. V2 - v0.9.zip` with the correct SHA-256 hash.

The steps below are **conceptual**; the attached code and manifests provide one concrete realization.

Step 1 – Obtain and verify the v0.9 archive

1. Obtain the file `A.R. V2 - v0.9.zip`.
2. Compute its SHA-256 hash.
3. Confirm that the hash matches the value given in Section 0.3.

If the hash matches, the archive is the canonical implementation bundle referred to here.

Step 2 – Inspect the manifest and core modules

1. Locate the **manifest file(s)** in the archive. These declare:
 - The feature alphabet (\mathcal{X}) and its tag schema.
 - Context-level band boundaries and hinge scales (UGM, temporal hinge).
 - Gate definitions and thresholds (Θ , κ , structural, CRA-like).
 - Shell structures and base strictness schedules for ParentGate.
 - Structural constants (e.g., (c) , unit scales for budgets).
2. Identify the **engine core modules**:
 - Data structures for site state (\mathcal{S}_k) (containing (W_k, Q_k) , budgets, tags, manifest/frame references).
 - Selector implementations.
 - Gate implementations (non-gravity gates and ParentGate).
 - Ratio-lexicographic ordering and fewest-acts tiebreak.
 - PF/Born ties-only module and RNG integration.
 - Budget computation and invariant-interval checks.

These modules collectively implement the algorithmic specification in Sections 4–6.

Step 3 – Implement the site state (\mathcal{S}_k)

To reconstruct the engine, define a site state structure with at least the following fields:

- (W_k) : finite world record (outer candidates, with context and shell tags).
- (Q_k) : finite qualia record (inner candidates, with modality, spatial, intensity, phase, and context tags).
- Budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$).
- Context-level and band tags ($-2 \dots +3$, roles, shell indices).
- Frame reference (CS/frame handle, if used).
- Manifest reference.
- Any internal bookkeeping fields (e.g., cached adjacency, RNG state) needed by the engine.

Ensure:

- (W_k) and (Q_k) are finite, as required by the finiteness axiom.
 - Context-level and shell tags are consistent with the CL mapping in Section 5.
-

Step 4 – Implement selectors and the hinge equality layer

1. Selectors

- Implement functions that, given (\mathcal{S}_k) , build a finite candidate set $(\mathcal{C}_k = \{(w_{k+1}^{(i)}, q_k^{(i)})\})$ by:
 - Enumerating local moves in (W_k) allowed by locality and no-skip,
 - Pairing them with inner candidates in (Q_k) that represent corresponding inner changes.

2. Feature maps and hinge equality

- Implement the manifest-declared feature maps:
[
 $f_k : W_k \rightarrow \mathbb{X}_i, \text{quad } g_k : Q_k \rightarrow \mathbb{X}_i.$
]
○ Filter (\mathcal{C}_k) to a hinge-admissible set $(\mathcal{C}_k^{\text{hinge}})$ by enforcing:
[
 $f_{k+1}(w_{k+1}) = g_k(q_k)$
]
(or the corresponding composite condition if multi-pixel tokens are used).
○ Discard any candidate that fails hinge equality.

This reproduces Steps 1–2 of the engine cycle (Section 4.3).

Step 5 – Implement non-gravity gates and ParentGate

1. Non-gravity gates (Θ, κ , structural, CRA-like)

- Implement each gate as a function from candidate and local state to a boolean (and optionally a small ordinal).
- Use only discrete tags, integer counts, and manifest parameters (no continuous control values) in accordance with the curve-ban.

2. ParentGate (gravity)

- Implement shell structures around containers, as declared in the manifest (shell indices, radial ranges, base strictness schedule).
- Implement ParentGate as a radial feasibility gate:
 - Identify which container and shells a candidate occupies or crosses,
 - Apply the effective strictness schedule (base schedule modulated by χ),

- Return pass/fail based on whether the candidate respects the shell-based feasibility constraints.

Apply these gates in the fixed order documented in Sections 4.4 and 6.2, producing a final feasible set ($\mathcal{C}_k^{\text{feas}}$).

Step 6 – Implement ordering, PF/Born ties-only, and commit

1. Ratio-lexicographic ordering

- For each candidate in ($\mathcal{C}_k^{\text{feas}}$), compute discrete residuals (fractions of failed checks) and form residual triples.
- Apply lexicographic ordering on these triples.

2. Fewest-acts tiebreak

- For candidates tied at the residual level, compute a discrete act-count or path-length measure.
- Restrict to those with minimal act count.

3. PF/Born ties-only

- If more than one candidate remains exactly tied:
 - Build an adjacency graph on the tie set based on local coherence rules (manifest-declared).
 - Construct a column-stochastic, primitive kernel (M).
 - Compute its Perron–Frobenius eigenvector (v).
 - Define probabilities ($p_i \propto v_i^2$) and normalize.
 - Use the RNG (with manifest-declared seeding) to select one winner.

4. Commit and budgets

- With the selected candidate ($c^{\text{ast}} = (w_{k+1}^{\text{ast}}, q_k^{\text{ast}})$):
 - Update (W_{k+1}) and (Q_{k+1}) accordingly.
 - Compute discrete flip counts and derive ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$).
 - Enforce the invariant interval ($\Delta t_k^2 = \Delta \tau_k^2 + \Delta x_k^2 / c^2$).
 - Form the next site (\mathcal{S}_{k+1}).

This reproduces Steps 3–7 of the engine cycle.

Step 7 – Use the provided drivers to reproduce v0.9 simulations

With the engine implemented as above:

1. Select a simulation family

- Open the V1 or V2 simulation document and choose a family (e.g., Q-series, matter-addition, T-series).

2. **Load the corresponding configuration**
 - Load the manifest and configuration files listed for that family (scene geometry, shell structure, run parameters).
 3. **Run the driver script or notebook**
 - Invoke the provided driver using your reconstructed engine.
 - Confirm that:
 - internal diagnostics (e.g., invariants, no-signalling checks) pass,
 - observable outputs (e.g., plots, tables) match or closely track those recorded in the v0.9 documentation.
-

Step 8 – Run audits to confirm structural compliance

Finally, use the **diagnostics and audit tools** from the archive (and/or re-implement them) to verify that:

- The engine respects finiteness, locality, and the no-skip rule.
- No continuous control (weights, soft activations) appears in the control path (curve-ban).
- PF/Born randomness is only invoked under the exact-tie condition.
- Typed budgets satisfy the invariant-interval relation for all accepted transitions.

If all audits pass and the simulation outputs reproduce the documented behaviours, the reconstructed engine can be considered a conforming realization of the present-act framework at the v0.9 level, as specified in this defensive publication.

9.3 Manifests and Parameter Tables

The **Manifest** is the central configuration object for any engine instance. It is part of the *theory*, not just a runtime settings file. Parameter tables derived from the manifest make explicit which values are fixed by the framework’s structure and which are left as engineering choices within allowed ranges.

9.3.1 Manifest Structure and Required Sections

A conforming manifest must, at minimum, define the following sections:

1. **Feature Alphabet Section**
 - Full definition of the finite feature alphabet (\mathcal{X}_i):
 - List of all tokens ($x_i \in \mathcal{X}_i$).
 - Tag schema for each token (modality, spatial bin, intensity bin, phase bin, context/band tags, etc.).
 - Definitions of:
 - Outer feature maps ($f_k : W_k \rightarrow \mathcal{X}_i$).

- Inner feature maps ($g_k : Q_k \rightarrow \mathcal{X}_i$).
- 2. **Context-Level and Hinge Section**
 - Explicit mapping from abstract context levels $(-2, -1, 0, +1, +2, +3)$ to:
 - physical scale bands (length ranges),
 - roles (inner machinery, hinge, containers).
 - Hinge parameters:
 - spatial hinge band around (L_{UGM}) ,
 - temporal hinge band around (T^*) .
- 3. **Gate Definitions Section**
 - Complete specification of non-gravity gates $(\Theta, \kappa, \text{structural, CRA-like})$:
 - discrete predicates and thresholds used for each gate,
 - ordering of gate application,
 - any finite ordinal scores used in residual calculations.
- 4. **ParentGate and Gravity Section**
 - Shell definitions around each container type (e.g., Earth, star, galaxy):
 - number of shells, shell index ranges, radial boundaries.
 - Base strictness schedules $(\sigma_{\text{base}}(s))$ for each shell index.
 - Definition of how χ modulates (σ_{base}) into (σ_{eff}) .
 - Any container-specific horizon templates or cutoffs.
- 5. **Budget and Interval Section**
 - Base unit scales (x_*, t_*, τ_0) .
 - Value of (c) used in the discrete invariant interval.
 - Rules for computing discrete counts (N_τ, N_t, N_x) from accepted transitions.
- 6. **PF/Born and RNG Section**
 - Rules for building adjacency graphs on tie sets.
 - Rules for constructing primitive column-stochastic kernels (M) .
 - Choice of RNG algorithm and seeding scheme (e.g., based on tick index and hashed state).
- 7. **Diagnostics and Audit Section**
 - Which audits are required (curve-ban, diagnostics-leak, PF/Born integrity, locality, invariant-interval consistency).
 - Parameters or thresholds used in these audits.

All of these sections are **immutable** at runtime. Changing them produces a different manifest and therefore a different theory instance, not a learned state of the same engine.

9.3.2 Parameter Tables: Structural vs Free Engineering Choices

For clarity, parameters in the manifest can be grouped into:

1. **Structurally Fixed Parameters** – set by the theory’s core structure, not to be tuned per application:
 - **Qualitative structure of (\mathcal{X}_i) :**
 - Which tags exist (e.g., modality, band, phase) and how they combine.
 - **Context-level roles and hinge assignments:**

- The fact that 0 is the hinge level; $-2/-1$ inner machinery; $+1/+2/+3$ containers.
- **Form of χ :**
 - χ as a dimensionless function of $((L_{\text{UGM}}), R_{\text{oplus}}, R_{\text{obs}}))$ with the specified monomial form.
- **Presence and structure of ParentGate:**
 - ParentGate as the sole gravity gate, operating on shells with monotone strictness schedules.
- **Invariant-interval relation:**
 - $(\Delta t^2 = \Delta \tau^2 + \Delta x^2 / c^2)$ as the fundamental budget identity.

These are **not** intended to be changed from one run to the next; they define the framework itself.

2. **Constrained Numerical Parameters** – may vary within a restricted range, but must respect the theory’s structural constraints:
 - **Hinge bands:**
 - Numerical values for the UGM band (e.g., 0.1–0.2 mm) and temporal hinge band (e.g., tens–few hundred ms), within ranges supported by evidence.
 - **Shell boundaries and counts:**
 - Exact radii for shells around Earth, galaxies, cosmic shells, etc., chosen within physically reasonable ranges, but respecting monotonic shell ordering and container roles.
 - **Base strictness values:**
 - Integer or ordinal values for $(\sigma_{\text{base}}(s))$, subject to monotonic or near-monotonic patterns and container-specific constraints.

These can be adjusted for different modelling scenarios (e.g., different container types, different scales) but must not violate the qualitative structure.

3. **Free Engineering Parameters (Within Limits)** – can be chosen more freely, provided they do not break finiteness, locality, or curve-ban:
 - Grid sizes, graph shapes, or resolution scales used in simulations.
 - Number of sites, run lengths, sampling counts.
 - Detailed adjacency thresholds used in structural gates (as long as they remain finite and local).
 - Logging frequencies and diagnostic sampling rates.

These choices affect fidelity and performance but not the core logical structure.

9.3.3 Example Parameter Tables

For documentation and reproducibility, it is recommended (and in v0.9, done) to present the manifest content as **parameter tables**, for example:

- **CL Band Table**
 - Columns: Level ($-2 \dots +3$), role, nominal length range, notes.
- **Hinge Table**
 - Columns: parameter name ((L_{UGM}) , (T^*) , (c)), nominal value, allowed range, justification.
- **ParentGate Shell Table** (for each container type)
 - Columns: shell index (s), radial range, $(\sigma_{\text{base}}(s))$, role (near-surface, core, outer region, horizon-like), notes.
- **Gate Threshold Table**
 - Columns: gate name (Θ , κ , structural, CRA), feature or count, threshold, type (structural vs engineering).

These tables make it explicit:

- Which parts of an implementation are encoding **the theory's structure**, and
- Which are **tunable choices** for particular simulations or models, all within the theory's constraints.

9.3.4 Manifest Versioning

Finally, the manifest should carry its own:

- **Version identifier** (e.g., `manifest_v0.9_core`),
- **Hash** (to lock in its content),
- **Change log** (listing structural vs numerical changes between versions).

This allows:

- Exact reconstruction of which manifest was used for a given simulation family, and
- Clear identification of when a change constitutes a new theory instance (e.g., new χ form, new gate types) versus a new configuration of the same structural framework (e.g., different shell boundaries within the same pattern).

9.4 Diagnostics and Verification Tools

The v0.9 implementation includes a set of **diagnostic and verification tools** whose purpose is to confirm that an engine instance actually respects the structural constraints specified in this defensive publication. These tools are as much a part of the framework as the engine itself: they operationalize the finiteness, locality, curve-ban, and PF/Born integrity requirements.

9.4.1 Log Structure for Auditing

To make auditing possible, each engine run is required to produce a **structured log** of key events. At minimum, the log must record for each tick (k):

- The site index (k).
- A compact description or hash of the state (\mathcal{S}_k) (including references to (W_k) and (Q_k)).
- The set of candidates (\mathcal{C}_k) and, for each candidate:
 - hinge-equality result,
 - gate outcomes (Θ , κ , structural, CRA-like, ParentGate),
 - residual triple and fewest-acts value.
- Whether PF/Born ties-only was invoked; if so:
 - the tie set (\mathcal{T}),
 - the adjacency graph structure on (\mathcal{T}),
 - the kernel (M) (or its defining data),
 - the PF eigenvector (v),
 - the chosen candidate and RNG seed/variates.
- The computed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$) and whether the invariant interval was satisfied.
- Any shell indices and container IDs relevant for ParentGate decisions.

Logs may be compressed (e.g., via hashes and summary statistics) in long runs, but they must contain enough information for the audits described below to be performed.

9.4.2 Curve-Ban Audit

The **curve-ban audit** checks that no continuous-valued control mechanism has been introduced into the engine's decision path.

Inputs:

- Engine source code and configuration.
- Logs from representative runs.

Checks:

- **Static code analysis:**
 - Scan control-path modules (selectors, gates, ratio-lex, fewest-acts, PF/Born) for:
 - floating-point weight arrays used in decisions,
 - gradient-based update rules,
 - continuous activation functions used in control (e.g., sigmoid, tanh, ReLU),

- external ML libraries used in the control path.
- Confirm that all decisions are based on discrete predicates and integer/ratio computations.
- **Runtime checks (optional):**
 - Monitor numerical ranges of intermediate control variables in logs.
 - Flag any occurrences of non-rational, non-integer parameters being used in gate thresholds or ordering rules.

Output:

- A report indicating:
 - Whether any violations were found,
 - Locations in code or logs where continuous control appears,
 - Recommended corrections (if any).

The engine passes the curve-ban audit if no continuous control mechanisms are detected in the decision path. Continuous quantities may still appear in diagnostics, but the audit must confirm that they do not influence control.

9.4.3 Diagnostics-Leak Audit

The **diagnostics-leak audit** ensures that data generated for **observation** or **analysis** does not leak back into the engine's control logic.

Inputs:

- Source code modules for:
 - engine control path,
 - diagnostic/logging functions.
- Logs with both control-path events and diagnostic outputs.

Checks:

- **Data-flow analysis:**
 - Trace data dependencies to ensure that diagnostic functions (e.g., those producing plots, computing continuous best-fit curves, or estimating derived quantities) do not feed their outputs into:
 - selectors,
 - gates,
 - acceptance rules,
 - or any control-path decisions.
- **API separation checks:**
 - Confirm that logging and diagnostics use read-only views of the engine state (or copies) and do not pass any derived data back into core engine modules.

Output:

- A report listing:
 - Any detected paths from diagnostic outputs into control logic,
 - Confirmation when no such paths exist.

The engine passes this audit when diagnostics are strictly one-way: from engine state to external logs, never from diagnostics back into decisions.

9.4.4 PF/Born Integrity Audit

The **PF/Born integrity audit** checks that:

- The PF/Born ties-only mechanism is invoked **only** under the exact-tie condition, and
- It is implemented exactly as specified (adjacency \rightarrow primitive kernel \rightarrow PF eigenvector \rightarrow squared weights \rightarrow RNG sampling).

Inputs:

- Engine code for PF/Born module.
- Logs indicating when PF/Born was invoked, including tie sets, kernels, eigenvectors, and RNG calls.

Checks:

- **Invocation condition:**
 - For each invocation recorded in the log, verify that:
 - the residual triples of candidates are identical (within the manifest's zeroing tolerance),
 - the fewest-acts values are identical,
 - the tie set size is $(|\mathcal{T}| \geq 2)$.
 - Confirm that no PF/Born calls occur when a unique candidate exists after deterministic ordering.
- **Kernel construction:**
 - Confirm that kernels (M) are:
 - column-stochastic ($\sum_i M_{ij} = 1$ for all (j)),
 - primitive (some power of (M) has strictly positive entries),
 - constructed from adjacency and discrete rules only.
- **Eigenvector and weights:**
 - Check that the eigenvector (v) is actually a (possibly approximated) PF eigenvector of (M) .
 - Check that probabilities recorded in logs satisfy $(p_i \propto v_i^2)$.
- **RNG usage:**
 - Confirm that RNG seeds and calls used for PF/Born are derived only from:
 - manifest-declared base seeds,

- local state identifiers (e.g., tick index, hashed state),
- not from external sources or diagnostics.

Output:

- A summary of all PF/Born invocations, including confirmation that:
 - each was triggered under exact-tie conditions,
 - kernels and weights were consistent with the specified procedure,
 - no extraneous randomness was introduced elsewhere in the control path.

9.4.5 Locality, No-Skip, and Invariant-Interval Audits

Additional core audits include:

- **Locality / No-Skip Audit:**
 - Using logs, verify that:
 - each engine step increments (k) by exactly 1,
 - changes in configuration between (W_k) and (W_{k+1}) are local in the sense defined by the manifest (bounded neighbourhoods, no jumps across the grid/graph).
- **Invariant-Interval Audit:**
 - For each accepted transition ($k \rightarrow k+1$), recompute:
 - discrete counts (N_{τ}, N_x, N_t) and budgets ($(\Delta \tau_k, \Delta x_k, \Delta t_k)$),
 - check that $(\Delta t_k)^2 = (\Delta \tau_k)^2 + (\Delta x_k)^2 / c^2$ holds within manifest-declared tolerances.
- **ParentGate Symmetry Audit:**
 - For containers meant to be approximately symmetric (e.g., spherical), test:
 - whether acceptance rates for candidates at the same shell index but different angular positions are statistically consistent,
 - whether any residual anisotropy is within expected discretization error.

Each audit produces a report that can be used to certify that an implementation satisfies the formal constraints on locality, interval structure, and gravity implementation.

9.4.6 Use of Diagnostic Tools in Practice

In the v0.9 workflow, these diagnostics and audits are used to:

- **Validate new engine builds or manifest changes:**
 - Any change to core engine code or to structural manifest sections is followed by rerunning audits on a standard set of test scenes.
- **Validate simulation results:**

- Before interpreting simulation outputs, audit logs are checked to confirm that:
 - no new violations appeared,
 - invariants and constraints were respected.
- **Provide a reproducibility baseline:**
 - The combination of:
 - a hashed archive,
 - a manifest with its own version and hash,
 - and a set of audit reports,

allows later investigators to verify that a given set of results was obtained from an engine that genuinely respects the present-act framework’s structural constraints.

Together, these diagnostic and verification tools make the finiteness, locality, curve-ban, PF/Born integrity, and gravity-implementation constraints **operational**, ensuring that the engine is not just conceptually but also practically aligned with the framework described in this defensive publication.

10. Terminology, Notation, and Accessibility

10.1 Glossary of Core Terms

This glossary collects the core terms used throughout the framework, in one place, for ease of reference.

Absolute Relativity / Overall V2 / Present-Act Framework

Different names for the same unified theory described in this document: a qualia-first, present-act framework with a formal core (V1), a discrete engine (V2), and a context-level ladder (CL).

Present-Act

The fundamental unit of reality in this framework: a single, discrete act of experiencing, taken as a finite whole. Each Present-Act has an inner (qualia) side and an outer (world) side.

Present-Moment Sphere (PMS)

The local state object for a single present: a structured triple $((\mathrm{IN}, \mathrm{ON}, \mathrm{CS}))$ representing the inner record, outer potential, and any shared collective context for a vantage at one tick.

Inner Network (IN)

The part of a PMS that stores committed record: everything that is already fixed for the present stream. Monotonically non-decreasing along admissible evolutions.

Outer Network (ON)

The part of a PMS that encodes structured potential: admissible candidate continuations of the current present. It represents the “open” side of the present before an act is committed.

Collective Sphere (CS)

A shared PMS-like structure that multiple vantages reference. It encodes common record and potential for a group (e.g., observers and apparatus) and supports frames and shared “objective” outcomes.

Stream (Subject)

A chain of Present-Acts ordered by a compatibility relation. What we ordinarily call a “subject” or “observer” is identified with such a stream (plus an indexing pattern), not with a separate metaphysical entity.

Context Level (CL) / Context Ladder

A hierarchical indexing of roles for parts of a situation relative to a Center. In this work, a six-band ladder is used: -2 (nanoband), -1 (micron/cellular), 0 (UGM/organism hinge), $+1$ (Earth-surface), $+2$ (galactic), $+3$ (cosmic shell).

Hinge (Level 0)

The central context level (0) where inner machinery and outer containers balance and where present-acts are centered. Spatially anchored at the UGM band; temporally anchored at the temporal hinge.

Universal Geometric Mean (UGM), ($L_{\{\text{UGM}\}}$)

A characteristic spatial hinge scale defined as the geometric mean of an inner cutoff (Planck-like) length and an outer cutoff (cosmic shell) length. Numerically, it lies in a band around ~ 0.1 – 0.2 mm and is used as the spatial “present pixel” at Level 0 .

Temporal Hinge, (T^*)

A characteristic temporal hinge scale representing the integration window of a single present-act at Level 0 (specious present). For human-scale systems, typically on the order of ~ 0.1 s.

Tick-State Carrier, (\mathcal{C}_k)

The formal V1 state at tick (k): ($\mathcal{C}_k = (k, h_k, \mathrm{IN}_k, \mathrm{ON}_k)$), where (h_k) is an abstract state, and ($\mathrm{IN}_k, \mathrm{ON}_k$) are inner and outer networks.

Primitive Operators (F, S, T, C, CT)

The basic V1 operations on carriers:

- F (Renew), S (Sink), T (Trade), C (Sync), CT (Framing).
Finite compositions of these generate the tick algebra and all admissible formal evolutions.

Hinge Projection Operator (historically “division-by-zero operator”)

A structural V1 operator that localizes context-time action to a hinge region and promotes it to a field-like description in a neighbourhood. No numerical division by zero is performed; the older name is treated as a historical nickname only.

Present Plane, (\mathcal{P})

A 2D real vector space with a complex structure (J), attached to the hinge. It supports present amplitudes and interference in a minimal way, without introducing a full Hilbert space.

Feature Alphabet, (\mathcal{X}_i)

A finite set of feature tokens used to encode both inner (qualia) and outer (world) content. Each token is a discrete bundle of tags (modality, spatial bin, intensity bin, phase bin, context/band tags, etc.).

World Record, (W_k)

The engine's finite representation of the outer side of the present at site (k): a set of world candidates, each tagged for feature extraction and context.

Qualia Record, (Q_k)

The engine's finite representation of the inner side of the present at site (k): a set of qualia candidates (qualia-pixels), each mapped into (\mathcal{X}_i) by (g_k).

Feature Maps, (f_k) and (g_k)

Manifest-declared maps ($f_k : W_k \rightarrow \mathcal{X}_i$) and ($g_k : Q_k \rightarrow \mathcal{X}_i$), used to encode world and qualia content in the feature alphabet and to enforce hinge equality.

Hinge Equality

The engine-level PMS boundary rule: a candidate pair ((w_{k+1}, q_k)) is admissible at the hinge only if ($f_{k+1}(w_{k+1}) = g_k(q_k)$) (possibly extended to small finite tuples). No metric closeness is used; only exact equality in (\mathcal{X}_i).

Typed Budgets ($(\Delta \tau, \Delta t, \Delta x)$)

Engine-level read-out quantities derived from discrete flip counts, representing inner-time-like, outer-time-like, and space-like changes. They are constrained by the discrete invariant interval ($\Delta t^2 = \Delta \tau^2 + \Delta x^2 / c^2$).

ParentGate

The dedicated gravity gate in V2.1. It applies a shell-dependent strictness schedule around containers (e.g., Earth, galaxies) to implement gravity as feasibility geometry. ParentGate is the only gate that depends directly on shells and χ .

Shells

Discrete radial zones around a container (e.g., (S_1, S_2, \dots, S_N)), each with a base strictness level. Used by ParentGate to define feasibility gradients and horizon-like behaviour.

Gravitational Amplitude, χ

A dimensionless amplitude constructed from context-ladder scales (UGM, container radius, cosmic shell), e.g. ($\chi = R_{\text{oplus}}^2 / (L_{\text{UGM}} R_{\text{obs}})$). It modulates ParentGate strictness and sets the overall strength of gravity-as-feasibility around a container.

Present-Act Engine (V2 / V2.1)

The discrete engine that realizes V1 under finiteness, locality, and curve-ban constraints: sites indexed by (k) , with (W_k, Q_k) , (χ_i) , gates, budgets, ParentGate, PF/Born ties-only, and a manifest-plus-audit structure.

Manifest

An immutable configuration object that is part of the theory: defines (χ_i) , feature maps, band boundaries, hinge scales, gate parameters, shell structures, χ usage, budgets, and audit rules. Changing the manifest (beyond allowed ranges) defines a new theory instance.

PF/Born Ties-Only

The engine's stochastic tie-resolution rule: when deterministic ordering fails to choose a unique candidate, a finite adjacency graph and primitive column-stochastic kernel (M) are built on the tie set, the PF eigenvector (v) is computed, and probabilities are assigned proportional to (v_i^2) . This is the only source of randomness and is used only under exact-tie conditions.

Curve-Ban

The requirement that the engine's control path use no continuous control functions (e.g., no floating-point weight fields, soft activations, or gradient descent). Decisions must be based on discrete predicates and integer/ratio computations.

No-Skip Rule

The locality condition that the engine advances only in single-tick steps $((k \rightarrow k+1))$ and only via local changes in configuration. No multi-tick jumps or nonlocal updates are permitted.

Context-Resolved Admissibility (CRA-like Gates)

Gates that enforce that candidates respect context partitions (lanes, families, or IDs). They prevent unintended merging of distinct contexts (e.g., different measurement setups or environment lanes) in a single act.

Frame

A CS equipped with a specific invariant-interval structure and synchronization conditions. Frames provide the discrete analogue of reference frames in relativity and measurement contexts in quantum theory.

Measurement-Induced Context Collapse (MICC-style Gates)

Specialized structural/CRA-like gates used in measurement setups to ensure that pointer-compatible basins are kept distinct and that, once a basin is selected, it is recorded consistently in $(\mathrm{IN}_{\text{CS}})$.

These terms and their roles are defined more fully in the preceding sections; this glossary is intended as a quick reference for readers navigating the full defensive publication.

10.2 Operator Names and Historical Labels

This subsection records a few naming conventions and historical labels so that different documents and versions of the framework can be read consistently. Where older or informal names exist, they are mapped to the preferred formal names used in this defensive publication.

Primitive V1 operators

The primitive operators of the V1 tick algebra appear in some earlier notes under slightly different labels. The mapping is:

- **F – Renew**
 - Historical / informal: sometimes “Refresh” or “Future-fill”.
 - Formal role: refreshes and extends the Outer Network (ON), generating or updating potential without erasing committed record.
- **S – Sink**
 - Historical / informal: “Commit”, “Collapse”.
 - Formal role: moves selected content from ON into IN, implementing the one-way flow from potential to record (the formal collapse operation).
- **T – Trade**
 - Historical / informal: “Exchange”, occasionally “Shuffle”.
 - Formal role: reallocates or exchanges content between parts of IN and/or ON under ledger and structural constraints, without changing total capacity.
- **C – Sync**
 - Historical / informal: “Synchronize”, sometimes “Align”.
 - Formal role: synchronizes PMSs within a Collective Sphere, aligning their shared IN/ON content and ensuring consistency across a frame.
- **CT – Framing**
 - Historical / informal: “Context-Set”, “Measurement Frame”, sometimes just “Frame”.
 - Formal role: establishes or modifies a frame on a CS, including which variables count as pointer states and which basins define outcome channels.

These five names (Renew, Sink, Trade, Sync, Framing) are the preferred operator labels in this defensive publication.

Hinge Projection Operator (historically “division-by-zero operator”)

One particular operator appeared in earlier philosophical and informal texts under the name “division-by-zero operator”. In this defensive publication it is referred to as the:

- **Hinge Projection Operator**

with the following clarification:

- The older “division-by-zero” name was metaphorical, intended to evoke a mapping from “infinite context” to a finite localized slice.
- No arithmetic division by zero is ever performed; the operator is a structural mapping that localizes context-time action to a hinge region and promotes it to a field-like description.
- All technical references in this document and in the v0.9 bundle should treat “division-by-zero operator” and “Hinge Projection Operator” as referring to the same formal construct, with the latter as the preferred term.

Engine and gate names

For completeness, the following engine-level terms also have fixed meanings:

- **Present-Act Engine (V2 / V2.1)**
 - Sometimes shortened to “engine” or “present-act engine”.
 - Refers to the discrete implementation with sites (k), records $((W_k, Q_k), (\chi_i))$, gates, budgets, ParentGate, and PF/Born ties-only.
- **ParentGate**
 - Historical / informal: “parent shell gate”, “gravity gate”.
 - Formal role: the unique gravity-specific feasibility gate that operates on shell indices around containers, using χ to modulate strictness.
- **CRA-like gates**
 - Historical / informal: “context-lane gates”, “lane separation”.
 - Formal role: Context-Resolved Admissibility gates that ensure candidates respect context partitions and prevent unintended merging of distinct context families.
- **MICC-style gates**
 - Historical / informal: “measurement collapse gates”, “pointer lanes”.
 - Formal role: Measurement-Induced Context Collapse gates that enforce pointer-compatible lanes in CS-framed measurement setups.

Where older documents use any of the historical labels above, they are to be understood as referring to the formal names and roles given here.

10.3 Symbol Index

This subsection lists the main symbols used in the framework and their meanings. It is not exhaustive of all notation in the attached documents, but it covers the core symbols that recur throughout this defensive publication.

Indices and Labels

- $(k \in \mathbb{Z})$
Discrete site / tick index in the engine and in V1 carriers. A single step ($k \rightarrow k+1$) corresponds to one present-act update.
 - $(n \in \mathbb{Z})$
Context-level index in the abstract ladder. In this work, the physically relevant band is $(n \in \{-2, -1, 0, +1, +2, +3\})$.
 - (\mathcal{L}_n)
Context level at index (n), e.g. (\mathcal{L}_0) (*hinge*), (\mathcal{L}_{+1}) (Earth band), etc.
 - $(s \in \{1, \dots, N\})$
Shell index for ParentGate around a container (radial shell label).
-

State Objects and Sets

- (\mathcal{C}_k)
Tick-State Carrier at tick (k) in V1: $(\mathcal{C}_k = (k, h_k, \mathrm{IN}_k, \mathrm{ON}_k))$.
 - (IN_k)
Inner Network at tick (k): committed record for the vantage.
 - (ON_k)
Outer Network at tick (k): admissible potential for the vantage.
 - $(\mathrm{IN}\{\text{CS}\}, \mathrm{ON}\{\text{CS}\})$
Shared inner/outer components for a Collective Sphere (CS).
 - (\mathcal{S}_k)
Engine site/state at tick (k) in V2: includes (W_k, Q_k) , budgets, tags, manifest/frame refs, and any internal bookkeeping.
 - (W_k)
World record at engine site (k): finite set (or multiset) of world candidates (w).
 - (Q_k)
Qualia record at engine site (k): finite set (or multiset) of qualia candidates (q).
 - $(\mathcal{C}_k, \mathcal{C}_k^{\{\text{hinge}\}}, \mathcal{C}_k^{\{\text{feas}\}})$
Candidate sets at site (k): raw candidates, hinge-admissible candidates, and fully gated (feasible) candidates, respectively.
 - (\mathcal{T})
Tie set passed into the PF/Born ties-only procedure when deterministic ordering fails to select a unique candidate.
-

Feature Alphabet and Maps

- (\mathcal{X}_i)
Finite feature alphabet. Each $(x_i \in \mathcal{X}_i)$ is a discrete token encoding modality, spatial bin, intensity bin, phase bin, context tags, etc.

- $(f_k : W_k \rightarrow \mathcal{X})$
Outer feature map at site (k): maps world candidates (w) to feature tokens in (\mathcal{X}) .
 - $(g_k : Q_k \rightarrow \mathcal{X})$
Inner feature map at site (k): maps qualia candidates (q) to feature tokens in (\mathcal{X}) .
 - $(\xi_{\text{out}}, \xi_{\text{in}})$
Shorthand for $(f_{k+1}(w_{k+1}))$ and $(g_k(q_k))$ in hinge tests.
-

Operators and Present-Plane Objects

- (F, S, T, C, CT)
Primitive V1 operators: Renew, Sink, Trade, Sync, Framing.
 - (J)
Complex structure on the Present Plane (\mathcal{P}) (satisfies $J^2 = -\mathrm{id}$).
 - \mathcal{P}
Present Plane: 2D real vector space with complex structure, used to represent present amplitudes.
 - (v_i)
Present-plane amplitude associated with a basin (R_i) on (IN) in V1.
 - (a_i)
Complex-number representation of (v_i) under an identification $\mathcal{P} \cong \mathbb{C}$; used in the structural Born rule.
 - (R_i)
Outcome basin on (IN) , corresponding to a distinct context-resolved record (e.g. a pointer state).
-

Ledger Functions and Budgets

- $(I(\mathcal{C}_k), E(\mathcal{C}_k), K(\mathcal{C}_k))$
Ledger functions on carriers: record budget (I), exposure (potential) budget (E), and capacity (K), satisfying $I_k + E_k = K_k$.
- (I_k, E_k, K_k)
Shorthand for $(I(\mathcal{C}_k), E(\mathcal{C}_k), K(\mathcal{C}_k))$.
- (N_τ, N_x, N_t)
Discrete flip counts along a segment: record-advancing flips, configuration-change flips, and tick count respectively.
- $(\Delta \tau, \Delta x, \Delta t)$
Typed budgets derived from flip counts:
[

$$\Delta \tau = N_\tau \tau, \quad$$

$$\Delta x = N_x x, \quad$$

$$\Delta t = N_t t.$$
]

- (τ^*, x^*, t^*)
Base unit scales for proper-time-like, space-like, and time-like budgets at the micro level.
 - (c)
Characteristic speed constant in the discrete invariant interval:
[
 $\Delta t^2 = \Delta \tau^2 + \frac{\Delta x^2}{c^2}$.
]
-

Dimensions, Pivot, and Ladder

- (D_k)
Effective inner dimension of (IN_k) (e.g., a fractal-dimension proxy).
 - $(D(n))$
Effective inner dimension at context level (\mathcal{L}_n) .
 - $(D_{\{\text{mem}\}}(n))$
Memory dimension at context level (\mathcal{L}_n) : how much of the inner structure is available as active record.
 - (D_{ast})
Hinge dimension, typically near 2, where inner geometry is optimally sheet-like for hinge behaviour.
 - $(g(D))$
Pivot function from dimension (D) to a scalar pivot value; normalized so $(g(D_{\text{ast}}) = 1)$.
-

Scales and Context-Level Lengths

- $(L_{\{\min\}}, L_{\{\max\}})$
Inner and outer cutoff scales used in defining the UGM (e.g., Planck-like length and cosmic shell scale).
- $(L_{\{\text{UGM}\}})$
Universal Geometric Mean (UGM) length:
[
 $L_{\{\text{UGM}\}} = \sqrt{L_{\{\min\}} L_{\{\max\}}}$
]
and, in application, $(\sqrt{\ell_{\{\text{P}\}} R_{\{\text{obs}\}}})$.
- (T^*)
Temporal hinge scale (specious-present timescale) at Level 0.
- (R_{\oplus})
Characteristic radius of a Level +1 container (e.g., Earth radius).
- $(R_{\{\text{obs}\}})$
Effective radius of the outermost context shell (Level +3), representing the observable cosmic shell or horizon-like scale.

Gravity and Shell Parameters

- (χ)
Gravitational amplitude derived from ladder scales. In v0.9:
[
$$\chi = \frac{R_{\text{oplus}}^2}{L_{\text{UGM}} R_{\text{obs}}}.$$

]
 - $(\sigma_{\text{base}}(s))$
Base strictness level for shell (S_s) in ParentGate.
 - $(\sigma_{\text{eff}}(s))$
Effective strictness level for shell (S_s), after modulation by χ and any container-specific rules:
[
$$\sigma_{\text{eff}}(s) = f(\sigma_{\text{base}}(s), \chi).$$

]
 - (S_s)
Shell (s) in the radial decomposition around a container, used by ParentGate.
-

PF/Born and Probabilities

- (M)
Primitive column-stochastic kernel on a tie set (\mathcal{T}), used in PF/Born ties-only.
 - (v)
Perron–Frobenius eigenvector of (M) : $(M v = \lambda_{\text{PF}} v)$.
 - (λ_{PF})
Perron–Frobenius eigenvalue (spectral radius) of (M) .
 - (w_i)
Intermediate weights in PF/Born ties-only: $(w_i \propto v_i^2)$.
 - (p_i)
Final normalized probabilities for candidates in (\mathcal{T}):
[
$$p_i = \frac{w_i}{\sum_j w_j}.$$

]
-

Residuals and Ordering

- $(\mathbf{d}(c))$
Residual triple for candidate (c):
[
$$\mathbf{d}(c) = (d_{\text{out}}(c), d_{\text{in}}(c), d_{\text{times}}(c))$$

]

-]
 - representing fractions of failed checks in outward, inward, and cross channels.
- (A(c))
Fewest-acts or path-length measure for candidate (c), used as a discrete tie-break after residual comparison.

These symbols, together with the terms defined in the glossary (Section 10.1), provide the core notation for the present-act framework as described in this defensive publication.

11. Explicit Claim Set and Failure Modes

11.1 Structural Claims

This subsection records the **structural claims** of the framework—statements that are asserted at the level of ontology and formal architecture, independently of any particular engine implementation or dataset. These claims define what counts as “the same theory” even if implementations or numerical choices vary within allowed ranges.

11.1.1 Ontological and Relational Structure

1. **Present-Acts as Primitives**
 - The fundamental units of reality are discrete Present-Acts: finite, indivisible acts of experiencing with inner (qualia) and outer (world) sides.
 - There is no deeper non-experiential “stuff” beneath present-acts; any effective “matter” or “field” is a pattern in the relations among present-acts.
2. **Pure Relativity**
 - All fundamental structure is given by relations among Present-Acts and their qualitative content.
 - There are no standalone substances with intrinsic properties independent of relational structure.
 - The “Subject” is identified with a stream (chain) of Present-Acts, not with a separate metaphysical entity.
3. **Infinite Present and Finite Slices**
 - There exists a single Infinite Present: the fully connected relational whole containing all possible global configurations.
 - Finite PMSs are local slices or versions of this whole, indexed into an ordered structure (time) by a compatibility relation (nested inclusion).
4. **IN/ON/CS Decomposition**
 - Each Present-Moment Sphere is decomposed into IN (committed record), ON (structured potential), and CS (shared collective context).
 - Every piece of local structure relevant to a present-act must appear either as committed record in IN or as admissible potential in ON, not in a third reservoir.

11.1.2 Formal Core (V1) Structure

5. Tick-State Carriers and Operator Algebra

- Formal states at each tick (k) are carried by Tick-State Carriers ($\mathcal{C}_k = (k, h_k, \mathrm{IN}_k, \mathrm{ON}_k)$).
- Evolution is generated by a finite primitive operator set ($\{F, S, T, C, CT\}$) acting on carriers, forming a tick algebra under composition.

6. Ledger and Arrow of Time

- Each carrier is equipped with a ledger $((I_k, E_k, K_k))$ satisfying $(I_k + E_k = K_k)$, with capacity (K_k) conserved under all primitive operators.
- Record budget (I_k) is monotone non-decreasing along admissible evolutions, establishing an intrinsic arrow of time.

7. Discrete Invariant Interval

- The theory defines integer flip counts $((N_\tau, N_x, N_t))$ and corresponding typed budgets $((\Delta \tau, \Delta x, \Delta t))$ such that:
[
$$\Delta t^2 = \Delta \tau^2 + \frac{\Delta x^2}{c^2}$$

]
holds for all admissible transitions.
- This relation defines a discrete Lorentz-like invariant interval and induces a cone structure limiting effective speeds to $(|\Delta x / \Delta t| \leq c)$.

8. Fractal Inner Geometry and Pivot Function

- The Inner Network has an effective inner dimension (D) in a bounded interval and a pivot function $(g(D))$ with a distinguished hinge dimension $(D_{\ast} \approx 2)$.
- The context ladder has a dimension profile $(D(n))$ and memory dimension $(D_{\text{mem}}(n))$; hinge behaviour occurs where $(D(n) \approx D_{\ast})$ and $(D_{\text{mem}}(n) \approx D(n))$.

9. Present Plane and Structural Born Rule

- A minimal Present Plane (\mathcal{P}) with complex structure (J) is attached to the hinge, supporting present amplitudes for IN basins.
- Under structural constraints (basin partitioning, context invariance, coarse-graining), outcome weights are proportional to squared amplitudes $(|a_i|^2)$ (structural Born rule).

10. Hinge Projection Operator

- A structural operator (Hinge Projection Operator) localizes context-time action to a hinge region and promotes it to a field-like description in a neighbourhood.
- This operator is purely structural; it does not involve arithmetic division by zero.

11.1.3 Context-Ladder, Hinge Scales, and Gravity-Related Structure

11. Six-Band Context Ladder

- A 6-band segment of the context ladder $(\{-2, -1, 0, +1, +2, +3\})$ is distinguished, with role assignments:
 - $-2, -1$: inner machinery (nano/biomolecular, micron/cellular),
 - 0 : hinge / organism scale (UGM band),
 - $+1$: local environment (Earth-surface band),

- +2: galactic environment (Milky Way–like disk band),
- +3: cosmic shell / horizon environment.

12. Universal Geometric Mean (UGM) Hinge

- The spatial hinge scale (L_{UGM}) is structurally defined as the geometric mean of inner and outer cutoff scales ((L_{\min}, L_{\max})):

$$L_{\text{UGM}} = \sqrt{L_{\min} L_{\max}}$$
 based on log-ladder structure and inner/outer role symmetry.
- For our universe, (L_{\min}) is identified with a Planck-like scale and (L_{\max}) with a cosmic shell scale, making (L_{UGM}) a hinge band around $\sim 0.1\text{--}0.2$ mm.

13. Temporal Hinge (T^*)

- There exists a temporal hinge (T^*) at Level 0: an integration window for a single present-act at the Center (specious present).
- (T^*) is an aggregate timescale (many micro-ticks), not a primitive tick; it is of order (~ 0.1) s for human-scale systems.

14. Gravity as Feasibility Geometry

- Gravity is encoded as **feasibility gradients** tied to the context ladder and pivot profile, not as a fundamental curvature field.
- Hinge-like layers at container boundaries (e.g., Level +1, +2) act as bias surfaces that make certain histories (near-surface, near-disk, near-shell) more feasible than others.

15. Gravitational Amplitude χ

- For a Level +1 container (e.g., Earth), a dimensionless gravitational amplitude χ is structurally defined using ladder scales:

$$\chi = \frac{R_{\text{oplus}}^2}{L_{\text{UGM}} R_{\text{obs}}}$$
 where (L_{UGM}), (R_{oplus}), and (R_{obs}) are the hinge, container, and cosmic shell scales, respectively.
- χ is fixed by structural constraints (dimensionless, correct monotonicities, use of ladder scales) and is not tuned per observable.

These structural claims specify the ontology and formal architecture that any conforming implementation of the framework must respect. Subsequent subsections (11.2–11.4) record engine-specific claims, empirical/phenomenological claims, and explicit failure modes and refuters.

11.2 Engine Claims

This subsection records the **engine-level claims**: what any implementation must do, at minimum, to count as an instance of the present-act engine described in this defensive publication.

11.2.1 Discrete Sites and Finite Records

1. Discrete site index

- The engine evolves over discrete sites ($k \in \mathbb{Z}$), with the fundamental update ($k \rightarrow k+1$) representing a single present-act.
- No multi-tick jumps or continuous time parameters are allowed in the control path.

2. Finite records at each site

- At each site (k), the engine maintains a finite site state (\mathcal{S}_k) containing:
 - A **world record** (W_k): a finite set (or multiset) of world candidates.
 - A **qualia record** (Q_k): a finite set (or multiset) of qualia candidates.
 - Typed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$).
 - Context-level tags, manifest and frame references, and any bounded internal bookkeeping.

3. Finite feature alphabet

- There exists a finite feature alphabet (\mathcal{X}).
- All inner and outer content relevant to control is encoded via manifest-defined maps ($f_k : W_k \rightarrow \mathcal{X}$) and ($g_k : Q_k \rightarrow \mathcal{X}$).

11.2.2 Hinge Equality and Candidate Construction

4. Local selectors

- Candidate transitions are built by selectors that:
 - Enumerate local moves in (W_k) and associated changes in (Q_k).
 - Respect locality and the no-skip rule.
- The raw candidate set (\mathcal{C}_k) is always finite.

5. Hinge equality as sole boundary rule

- For each candidate ((w_{k+1}, q_k)), hinge admissibility is determined by:
 - [
 $f_{k+1}(w_{k+1}) = g_k(q_k),$
]
 - possibly extended to small finite tuples.
- Only candidates satisfying this exact equality in (\mathcal{X}) are passed to later stages.
- No metric similarity, soft thresholds, or continuous scoring functions are used in hinge tests.

11.2.3 Gates and Feasibility Filtering

6. Non-gravity feasibility gates

- The engine applies a fixed sequence of non-gravity gates (Θ, κ , structural, CRA-like), each:
 - Defined entirely in terms of discrete tags and integer/ratio counts.
 - Returning boolean or finite-ordinal outcomes.
 - Independent of χ and large-scale shell structure.

7. ParentGate as the unique gravity gate

- Gravity is implemented solely via ParentGate, which:
 - Operates on discrete shell indices around containers.
 - Uses a manifest-declared base strictness schedule modulated by χ .
 - Returns a boolean pass/fail based on radial feasibility.
 - No other gate may depend directly on radial shells or χ .
- 8. Fixed gate order and separation of roles**
- Gates are applied in a fixed, manifest-declared order:
 - Hinge equality \rightarrow non-gravity gates \rightarrow ParentGate \rightarrow ordering and PF/Born (if needed).
 - Gravity-specific feasibility (ParentGate) is strictly separated from non-gravity gates.

11.2.4 Ordering, PF/Born Ties-Only, and Commit

- 9. Deterministic ordering before randomness**
- After gating, candidates are ordered by:
 - Ratio-lexicographic comparison of discrete residuals.
 - A fewest-acts (or path-length) tiebreak.
 - If this produces a unique winner, selection is deterministic.
- 10. PF/Born ties-only rule**
- Randomness is invoked only if:
 - Multiple candidates remain exactly tied after all deterministic ordering (same residuals and same fewest-acts value).
 - In that case:
 - An adjacency graph is built on the tie set using local, discrete rules.
 - A primitive column-stochastic kernel (M) is constructed.
 - The Perron–Frobenius eigenvector (v) of (M) is computed.
 - Probabilities are assigned proportional to (v_i^2) , and an RNG is used to select a winner.
 - No other source of randomness is permitted in the control path.
- 11. Commit and budgets under invariant interval**
- The selected candidate defines the next state (\mathcal{S}_{k+1}): *updated* (W_{k+1}), (Q_{k+1}), and context tags.
 - Typed budgets ($(\Delta \tau_k, \Delta t_k, \Delta x_k)$) are computed from discrete counts and must satisfy:

$$\left[\begin{array}{l} \Delta t_k^2 = \Delta \tau_k^2 + \frac{\Delta x_k^2}{c^2}. \end{array} \right]$$
 - Transitions violating this relation are not accepted.

11.2.5 Implementation Axioms: Finiteness, Locality, Curve-Ban, Audits

- 12. Finiteness and locality (no-skip)**
- All sets in the control path are finite.
 - Updates are strictly $(k \rightarrow k+1)$; no multi-tick jumps.

- Configuration changes are local in the sense defined by manifest-declared neighbourhoods.
- 13. Curve-ban (no continuous control)**
 - The control path uses only:
 - discrete predicates,
 - integer or rational ratios,
 - finite-ordinal comparisons.
 - No continuous-valued weight fields, soft activations, or gradient-based updates are allowed in selectors, gates, or ordering.
- 14. Manifest as part of the theory**
 - The manifest declares (χ), band mappings, gate definitions, shell structures, χ usage, budgets, and audits.
 - It is immutable at runtime; changing it (beyond allowed ranges) defines a new theory instance, not a learned state.
- 15. Audits and diagnostics**
 - Implementations must provide logs and tools sufficient to audit:
 - curve-ban compliance,
 - diagnostics-leak separation,
 - PF/Born ties-only integrity,
 - locality and no-skip behaviour,
 - invariant-interval consistency.
 - Passing these audits is part of the engine's definition, not an optional extra.

These engine claims specify what any conforming implementation of the present-act engine must do. If one or more of these conditions are violated, that implementation is not considered an instance of the framework as defined in this defensive publication.

11.3 Empirical and Phenomenological Claims

This subsection records the **empirical and phenomenological claims** the framework makes about the actual world, as supported (to varying degrees) by the v0.9 evidence and simulation suite. These go beyond purely structural or engine-level statements and are intended as concrete, testable assertions.

They are grouped by theme.

11.3.1 Context-Ladder and Hinge-Scale Claims

- 1. Existence of a six-band scale ladder in nature**
 - Physical systems exhibit a hierarchy of characteristic scales that naturally cluster into bands corresponding to:
 - a nanoband ($\approx 1\text{--}200\text{ nm}$),
 - a micron band ($\approx 0.2\text{--}50\text{ }\mu\text{m}$),
 - a hinge band near $\approx 0.1\text{--}0.2\text{ mm}$ (UGM band),

- an Earth-surface band ($\approx 1\text{--}100\text{ km}$),
 - a galactic disk band (kpc scale, Milky Way-like),
 - a cosmic shell band (horizon-like cosmological scales).
- Each band shows qualitatively different behavioural patterns (inner machinery, hinge, containers), consistent with the roles assigned in the CL ladder.
- 2. UGM band as a real hinge scale**
 - The geometric mean of a Planck-like inner scale and a cosmic shell scale lands in a narrow physical band around $\sim 0.1\text{--}0.2\text{ mm}$.
 - Across heterogeneous datasets (biological structures, materials, and other systems), there is non-trivial clustering of characteristic sizes near this UGM band, consistent with its role as a global hinge (Level 0).
- 3. Temporal hinge as a real integration window**
 - For human and similar nervous systems, there exists an empirical temporal band (on the order of $\sim 0.1\text{ s}$) that behaves as a present-act integration window:
 - many micro-events are integrated into a single, coherent “moment”,
 - changes significantly shorter than this window are not experienced as distinct events at the Center.
 - This band matches the temporal hinge (T^*) used in the framework’s CL and engine constraints.
- 4. CNS/organism size band from geometric means**
 - The geometric mean between UGM and Earth-surface scales yields a band that overlaps with observed size ranges of organisms with complex central nervous systems.
 - Characteristic CNS and body sizes cluster near this band more than would be expected if sizes were distributed without reference to such a geometric relationship.

11.3.2 Gravity and Feasibility Claims

- 5. χ as an order-of-magnitude match to weak-field GR amplitude**
 - The gravitational amplitude χ , defined structurally as

$$\chi = \frac{R_{\text{oplus}}^2}{L_{\text{UGM}} R_{\text{obs}}},$$
 falls in the same order-of-magnitude band as the weak-field GR metric potential amplitude at Earth’s surface (e.g., (r_s / R_{oplus})).
 - In particular, χ differs from the standard GR metric amplitude by an order-unity factor of ≈ 1.8 , which is explicitly acknowledged and treated as a first-pass prediction subject to refinement.
- 6. Gravity triad: deflection, delay, redshift from feasibility-only engine**
 - With χ fixed by the ladder scales, and without inserting explicit metric fields or potentials into the engine, the present-act engine, configured with ParentGate and the invariant interval, reproduces:
 - ray deflection angles in the correct order of magnitude and scaling with impact parameter,

- time-delay patterns (Shapiro-style) with approximately logarithmic dependence on impact parameter,
 - redshift/time-dilation-like behaviour across shells at the right scale.
 - All three appear under a single χ -family; no separate per-observable tuning is introduced.
- 7. Milky Way-scale activation at +2↔+3 seam**
- Lensing data binned by galaxy size and mass are better described (under standard model-selection criteria) by models that include an activation of lensing strength at or near a Milky Way-like disk radius than by strictly size-only models.
 - This activation scale aligns with the +2↔+3 seam in the CL ladder and is consistent with the framework's prediction that a Milky Way-scale band acts as a hinge for +2-level containers.
- 8. RAR-like behaviour from hinge and CL structure**
- When rotation-curve data are analysed using a hinge-based, flat-window method derived from the CL framework, the resulting radial acceleration relation (RAR) shows:
 - a low-acceleration regime where observed acceleration scales approximately as the square root of baryonic acceleration,
 - with a single global acceleration scale emerging from the same structural picture, rather than from per-galaxy fitting.
 - This is consistent with the framework's claim that context-ladder structure and hinge scales can produce RAR-like behaviour without dark-matter halos defined as independent continuous fields.

11.3.3 Quantum-Like Behaviour Claims

- 9. Quantum interference and complementarity from present-act engine**
- In controlled scenes (e.g., interferometer-like setups), the engine reproduces interference patterns in outcome frequencies using:
 - discrete adjacency and PF/Born ties-only,
 - no continuous wavefunctions in the control path.
 - When which-path information is encoded and protected in ($\backslash X_i$) and via CRA-like gates, interference is suppressed; when it is erased or not encoded, interference reappears, matching the complementarity pattern of standard quantum experiments.
- 10. Decoherence and effective classicality from environment coupling**
- When environment states and context tags are included in (W_k, Q_k), and CRA-like gates are configured appropriately, macroscopically distinct outcomes rapidly become separated into different context lanes.
 - In these regimes, the PF/Born step is invoked rarely or only on local micro-ties, and macroscopic behaviour appears effectively classical and deterministic, consistent with decoherence expectations.
- 11. No-signalling maintained in CHSH-type tests**
- In CHSH/Bell-type configurations constructed in the engine, the framework reproduces:

- quantum-like correlations that violate classical Bell inequalities,
- while maintaining strict locality and no-signalling (no control over marginal outcome probabilities at one site by choices made at a space-like separated site).
- This supports the claim that non-classical correlations can emerge from the present-act engine under PF/Born ties-only, without superluminal signalling channels.

11.3.4 Cross-Scale and Integrative Claims

12. Cross-scale coherence of hinge, ladder, and gravity

- The same context ladder and hinge scales that structure biology and perception (UGM band, temporal hinge, CNS size band) also:
 - enter into the definition of χ ,
 - determine where containers (e.g., galaxies) show activation effects,
 - and underlie the engine's gravity-as-feasibility behaviour.
- The framework therefore claims an integrated explanation in which:
 - biological, perceptual, and gravitational scales are not independent accidents but manifestations of the same underlying context-ladder structure.

13. Unified present-act description of SR, QM, and weak-field gravity

- With the engine configured according to the structural constraints, the same present-act dynamics:
 - respect a Lorentz-like invariant interval and no-skip locality (SR-like behaviour),
 - produce interference, decoherence, and Born-rule statistics (QM-like behaviour),
 - and yield deflection, delay, redshift, and activation patterns consistent with weak-field gravity at hinge scales (gravity-as-feasibility).
- This is claimed to be achieved without introducing separate, ad hoc force fields or potentials in the engine's core logic.

These empirical and phenomenological claims are supported to varying degrees by the simulations and CL probes recorded in the v0.9 bundle. They are presented here explicitly so that future work can confirm, refine, or refute them. Subsequent subsection 11.4 describes specific **failure modes and refuters** that would significantly weaken or falsify these claims.

11.4 Failure Modes and Potential Refuters

This subsection lists concrete conditions under which the framework, as defined in this defensive publication, would be **seriously weakened** or **effectively refuted**. These are stated so that future work (by the author or others) has clear targets for testing and potential falsification.

11.4.1 Structural / Engine-Level Refuters

1. Violation of the discrete invariant interval in a conforming engine

- If, in an implementation that otherwise passes finiteness, locality, and curve-ban audits, it is shown that:
 - accepted transitions systematically require budgets $((\Delta \tau, \Delta t, \Delta x))$ that cannot satisfy
$$\left[\begin{aligned} \Delta t^2 &= \Delta \tau^2 + \frac{\Delta x^2}{c^2} \end{aligned} \right]$$
under any consistent assignment of unit scales $((\tau_*, x_*, t_*))$ and (c) ,
 - and this persists even after correcting for coding errors or configuration mistakes,
- then the core structural link between the tick algebra and SR-like behaviour would be broken. This would refute the claim that the present-act engine can reproduce relativistic cone structure from its primitive operations.

2. Necessity of continuous control in the core path

- If it is shown that no engine satisfying:
 - finiteness,
 - locality and no-skip,
 - and a strict curve-ban (no continuous control in selectors/gates/ordering), can reproduce basic quantum-like phenomena (e.g., stable interference patterns, Born-rule statistics) or basic weak-field gravitational phenomena (e.g., correct deflection scaling),
- and that such behaviour only appears when continuous control (e.g., trainable weight fields, soft activations, gradient descent) is introduced into the **control path**,
- then the claim that SR, QM, and gravity can be realized in a purely discrete, combinatorial present-act engine would be strongly undermined.

3. Failure of PF/Born ties-only to reproduce Born statistics in simple scenes

- If, in well-controlled interference-like scenes where standard quantum mechanics predicts clear Born-rule distributions, careful long-run engine runs show that:
 - the PF/Born ties-only procedure, when correctly implemented and audited, yields stable frequencies that **significantly and reproducibly** deviate from $(|a_i|^2)$ in a way that cannot be attributed to discretization or finite sampling,
- then the claim that the structural Born rule is correctly realized by PF/Born ties-only would be challenged.

4. Unavoidable diagnostics leak or hidden continuous channels

- If it is demonstrated that any implementation that reproduces the reported behaviours must, in practice, rely on:
 - diagnostic outputs feeding back into control, or
 - hidden continuous parameter channels (e.g., unnoticed weight matrices, implicit learning),
- then the separation between control and diagnostics and the curve-ban posture would be invalidated.

11.4.2 Context-Ladder and Hinge-Scale Refuters

5. Systematic failure of UGM as a global hinge

- If high-quality, cross-domain datasets (biological, physical, materials, etc.) consistently show that:
 - there is **no statistically meaningful clustering** of characteristic scales near the $\sim 0.1\text{--}0.2$ mm band predicted by (L_{UGM}) , and
 - alternate hinge candidates (e.g., other means of $((L_{\min}, L_{\max}))$ or entirely different scale constructions) provide a clearly better explanation of observed clustering and hinge-like behaviour,
- then the claim that UGM is a real, meaningful hinge scale in nature would be weakened or refuted.

6. Breakdown of the six-band ladder as a useful structure

- If detailed, multi-domain analysis shows that:
 - the proposed bands $(-2, -1, 0, +1, +2, +3)$ and their seams **do not** correlate with any stable, interpretable clustering or plateau behaviour in real data,
 - and that any apparent clustering is easily explained by simpler, non-ladder-based models (e.g., smooth distributions or known scaling laws),
- then the CL ladder's empirical relevance would be called into question. The formal ladder could still exist, but its claim to correspond to real-world structure would be severely weakened.

7. Failure of cross-scale GM relationships

- If it is shown that geometric-mean relationships (e.g., Planck–Universe \rightarrow UGM, UGM–Earth \rightarrow CNS size band) provide no better explanation of observed scale distributions than arbitrary or random choices, and that statistically robust alternative relationships clearly outperform them,
- then the idea that these GM constructions capture real pivot structures would be undermined.

11.4.3 Gravity and χ Refuters

8. Persistent, large mismatch between χ -family and weak-field observables

- If, after:
 - refining the mapping from ladder indices to physical scales,
 - tightening UGM and (R_{obs}) estimates within reasonable cosmological ranges,
 - and improving container models (e.g., more realistic Earth, galaxy mass profiles),
- the χ -family derived from $(\chi = R_{\oplus}^2 / (L_{\text{UGM}} R_{\text{obs}}))$ still **systematically fails** to match:
 - weak-lensing deflections,
 - Shapiro delays,

- gravitational redshifts,
 - by factors larger than modest order-unity discrepancies (e.g., $> 3-5$) and in a way not fixable by reasonable structural refinements,
 - then the χ -construction as a universal hinge-based amplitude would be effectively refuted.
- 9. **Requirement for separate χ -like parameters per observable**
 - If it turns out that:
 - reproducing deflection, delay, and redshift simultaneously requires **independent tuning** of three separate amplitude-like parameters (or highly contrived χ modifications),
 - then the claim that a single χ -family, structurally derived from ladder scales, can underwrite all three would be invalidated.
- 10. **Failure of +2 \leftrightarrow +3 activation in more complete lensing datasets**
 - If, in larger and more precise galaxy–galaxy lensing datasets, model-selection tools consistently favour:
 - **strict size-only models** over any model that includes Milky Way–scale activation at +2 \leftrightarrow +3,
 - and this result is robust to changes in binning, sample selection, and reasonable model variations,
 - then the claim of a Milky Way–scale activation seam (as predicted by the CL ladder) would be severely weakened.
- 11. **Failure of hinge-based RAR explanation**
 - If careful analysis shows that:
 - the RAR-like behaviour reported in v0.9 is an artefact of selection or methodology, and
 - the hinge-based/gm-based construction does **not** robustly reproduce RAR features across independent, high-quality datasets,
 - while simpler or orthogonal models do,
 - then the CL-based explanation of RAR would be refuted.

11.4.4 Quantum and Measurement Refuters

12. Non-reproducible interference and complementarity patterns

- If independent implementations of the engine, or careful re-runs of the v0.9 code, fail to reproduce:
 - stable interference patterns in simple interferometer-like scenes, or
 - complementarity (which-path vs interference) as encoded via ($\setminus X_i$) and CRA-like gates,
- and these failures cannot be traced to implementation errors,
- then the claim that quantum-like interference and complementarity are structurally embedded in the engine would be weakened.

13. Demonstrated signalling in CHSH-type tests

- If, after exhaustive auditing, it is found that:
 - in CHSH/Bell-type scenes built according to the framework, one can **controllably signal** between space-like separated sites by adjusting settings,
- and this persists in correctly localized, no-skip configurations,
- then the claim that the engine maintains no-signalling while reproducing quantum-like correlations would be refuted.

11.4.5 Cross-Scale Integrative Refuters

14. Lack of cross-domain coherence

- If, after broader and deeper empirical work, it becomes clear that:
 - the scales that appear in biology (UGM band, CNS size band, temporal hinge) are **uncorrelated** with those that appear in gravity and cosmology (e.g., no consistent mapping between ladder-based GM scales and observed astrophysical hinges),
- then the claim that the same context-ladder structure underlies both biology/perception and gravity would be weakened.

15. Necessity of independent sectoral models

- If, in practice, achieving good fits to:
 - SR-like behaviour,
 - quantum-like behaviour, and
 - weak-field gravity requires three essentially **independent** modelling frameworks, each with its own free parameters and no shared structural core,
- then the claim of a unified present-act framework providing a common structural basis for all three would be effectively refuted.

These failure modes and potential refuters are stated explicitly so that the framework can be treated as **genuinely testable**. If one or more of these conditions are met under careful, independent investigation, it would significantly weaken or falsify the corresponding claims in Sections 11.1–11.3.