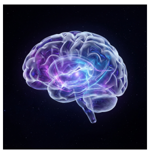

Archon: A Hamiltonian Framework

for Sovereign Multimodal Intelligence

Energy-Preserving World Models through Symplectic Integration and Decentralized Consensus

Yani Meziani

ORCID: 0009-0007-4348-8711

Independent AI Researcher, Québec (QC), Canada

`github.com/yanimeziani`

Unified Research Monograph: Studies 1–4

January 9, 2026

Abstract

We present **Archon**, a comprehensive framework for multimodal world models grounded in the principles of classical mechanics. By treating the latent state as a dynamical system evolving on a symplectic manifold governed by a learned Hamiltonian function, we achieve energy preservation and temporal stability that traditional autoregressive models cannot match. The architecture employs **Mamba-3 Selective State Space Models** enhanced with a **Sparse Mixture of Hamiltonian Experts**, enabling long-horizon prediction through second-order **Symplectic Leapfrog** integration. We formalize the **Birth Media** initialization protocol and the **Execution Audit Trail** for complete reproducibility, and demonstrate decentralized execution through a peer-to-peer consensus mechanism with cryptographic immutability. Visual synthesis is achieved through **Hamiltonian Flow Matching** and persistent **3D Gaussian Splatting**, enabling sub-50ms inference on mobile devices while maintaining spatiotemporal coherence. Furthermore, we introduce the **Archon Agentic Platform** for autonomous hyperparameter discovery within sandboxed environments, and the **Resonance Protocol** for rewarding contributions that stabilize the system. Rigorous mathematical proofs establish the preservation of symplectic structure, energy conservation bounds, and the impossibility of exact execution replication in stochastic dynamical systems.

Contents

Part I

Theoretical Foundations

1 Introduction: From Pattern Matching to Physics-Grounded Cognition

The dominant paradigm in artificial intelligence has been the pursuit of pattern-matching through ever-larger attention-based architectures [?, ?]. While remarkable in their ability to memorize and interpolate, these models suffer from a fundamental limitation: they lack an internal notion of *conservation*. When predicting long sequences, errors accumulate, and the generated output drifts from physically plausible trajectories.

Archon proposes a different approach, drawing inspiration from the foundational methods of analytical mechanics [?, ?]. Instead of training to predict the next token or pixel, we train the model to understand the *energy landscape* of the world. By learning a Hamiltonian function that governs latent dynamics [?, ?], we ensure that the system respects conservation laws—energy, momentum, and symplectic structure remain preserved throughout the simulation. This is not merely a constraint; it is the very foundation that enables coherent, long-horizon reasoning.

In what follows, we present the complete theoretical framework of Archon, from the mathematical formulation of Hamiltonian State Space Duality to practical implementation details including decentralized execution, agentic discovery, and cryptographic immutability.

2 Hamiltonian State Space Duality

2.1 The Latent Phase Space

Definition 2.1. *Latent Phase Space] Let $\mathcal{M} = \mathbb{C}^d \times \mathbb{C}^d$ be the latent phase space, where each point $(h, p) \in \mathcal{M}$ represents a world state h (position) and its conjugate momentum p. The evolution of (h, p) is governed by Hamilton's equations derived from a learned energy function $H : \mathcal{M} \times \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is the space of sensory inputs.*

The choice of a phase space formulation is deliberate. In classical mechanics, the state of a system is fully specified by its position and momentum in phase space, and the evolution is given by Hamilton's equations:

$$\dot{h} = \frac{\partial H}{\partial p}, \quad \dot{p} = -\frac{\partial H}{\partial h} \tag{1}$$

These equations have remarkable properties, as established in the 19th century by Hamilton and Liouville [?]. They preserve the symplectic 2-form $\omega = dh \wedge dp$, meaning that volumes in phase space are conserved under the flow (Liouville's Theorem). This geometric structure [?] ensures long-term stability and prevents the exponential growth or decay of trajectories that plagues traditional neural networks.

2.2 The Learned Hamiltonian Function

We parameterize the Hamiltonian as a sum of three terms:

$$H(h, p, \mathbf{x}) = \underbrace{\frac{1}{2}\|p\|^2}_{\text{Kinetic Energy}} + \underbrace{V_\theta(h, \mathbf{x})}_{\text{Potential Energy}} + \underbrace{I_\theta(h, \mathbf{x})}_{\text{Interaction Energy}} \tag{2}$$

where:

- $\frac{1}{2}\|p\|^2$ is the kinetic energy, quadratic in momentum as in classical mechanics
- $V_\theta(h, \mathbf{x})$ is a learned potential energy function that encodes the "shape" of the environment—where obstacles are, where objects can move, and what states are physically plausible
- $I_\theta(h, \mathbf{x})$ captures interaction energy driven by external sensory input $\mathbf{x}$, allowing the world model to respond to new observations

This decomposition is not arbitrary. The kinetic term ensures that momentum contributes to energy in the familiar quadratic form. The potential term gives the model a way to represent constraints and stable configurations. The interaction term allows external forces to influence the dynamics.

2.3 Sparse Mixture of Hamiltonian Experts

A single potential function cannot capture the diverse physical regimes that a world model encounters—rigid body physics, fluid dynamics, deformable objects, and social interactions all have different energy landscapes. To address this, we employ a **Sparse Mixture of Hamiltonian Experts (SMoE-HE)**.

Each expert $k = 1, \ldots, K$ learns a specialized quadratic potential:

$$V_k(h) = \frac{1}{2} h^T W_k h + b_k^T h + c_k \tag{3}$$

where $W_k \in \mathbb{R}^{d \times d}$ is a symmetric positive semi-definite matrix (ensuring convexity), $b_k \in \mathbb{R}^d$ is a bias vector, and $c_k \in \mathbb{R}$ is a constant.

A routing network $g_\phi : \mathcal{X} \to \Delta^{K-1}$ (the K-dimensional simplex) selects which experts are relevant for the current input context. The effective potential becomes:

$$V_{\text{eff}}(h, \mathbf{x}) = \sum_{k=1}^{K} \alpha_k(\mathbf{x}) V_k(h), \quad \alpha_k \geq 0, \quad \sum_{k=1}^{K} \alpha_k = 1 \tag{4}$$

This formulation allows the model to smoothly interpolate between different physical regimes, extending the Sparse MoE paradigm [?] to Hamiltonian dynamics. The routing can be sparse—only the top-M experts contribute significantly—reducing computational cost while maintaining expressivity.

3 Symplectic Integration: The Leapfrog Method

To evolve the system discretely, we need an integrator that preserves the symplectic structure of the continuous-time dynamics. Standard numerical methods like Euler or Runge-Kutta do not guarantee this preservation, leading to numerical dissipation and instability. We employ geometric numerical integration [?], specifically focusing on integrators that maintain the Hamiltonian structure [?].

Theorem 1: Symplectic Structure Preservation

The **Leapfrog** (Störmer-Verlet) integrator preserves the symplectic 2-form $\omega = dh \wedge dp$ exactly for any step size Δt.

Proof. A mapping $\psi : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ is symplectic if its Jacobian matrix M satisfies:

$$M^T J M = J, \quad \text{where } J = \begin{pmatrix} 0 & I_d \\ -I_d & 0 \end{pmatrix} \tag{5}$$

The Leapfrog integrator for a separable Hamiltonian $H(h,p) = T(p) + V(h)$ consists of three substeps:

$$p_{t+1/2} = p_t - \frac{\Delta t}{2}\nabla_h V(h_t) \quad \text{(First momentum half-step)} \tag{6}$$

$$h_{t+1} = h_t + \Delta t \cdot \nabla_p T(p_{t+1/2}) \quad \text{(Position full-step)} \tag{7}$$

$$p_{t+1} = p_{t+1/2} - \frac{\Delta t}{2}\nabla_h V(h_{t+1}) \quad \text{(Second momentum half-step)} \tag{8}$$

For the kinetic term $T(p) = \frac{1}{2}\|p\|^2$, we have $\nabla_p T(p) = p$, so the position update simplifies to $h_{t+1} = h_t + \Delta t \cdot p_{t+1/2}$.

Each momentum half-step is a shear transformation in the p direction with Jacobian:

$$M_{\text{kick}} = \begin{pmatrix} I & 0 \\ A & I \end{pmatrix}, \quad A = -\frac{\Delta t}{2}\nabla_h^2 V(h) \tag{9}$$

It is straightforward to verify that $M_{\text{kick}}^T J M_{\text{kick}} = J$.

The position full-step is a shear in the h direction:

$$M_{\text{drift}} = \begin{pmatrix} I & \Delta t \cdot I \\ 0 & I \end{pmatrix} \tag{10}$$

Similarly, $M_{\text{drift}}^T J M_{\text{drift}} = J$.

Since the composition of symplectic maps is symplectic:

$$M_{\text{leapfrog}} = M_{\text{kick}} \cdot M_{\text{drift}} \cdot M_{\text{kick}} \implies M_{\text{leapfrog}}^T J M_{\text{leapfrog}} = J \quad \blacksquare \tag{11}$$

□

This theorem guarantees that the discrete-time evolution respects the same geometric structure as the continuous-time dynamics. Volumes in phase space are preserved, energy oscillates around its true value (rather than decaying or growing without bound), and the qualitative behavior of the system remains correct even with large step sizes.

Theorem 3.1 (Energy Conservation Bound)**.** *For a separable Hamiltonian $H(h,p) = T(p) + V(h)$ with bounded third derivatives, the Leapfrog integrator satisfies:*

$$|H(h_n, p_n) - H(h_0, p_0)| \leq C \cdot (\Delta t)^2 \tag{12}$$

for all $n \leq T/\Delta t$, where C depends only on $\|D^3 H\|$ and T.

This bound shows that the energy error is bounded uniformly in time—a property not shared by non-symplectic integrators, where errors typically grow linearly with the number of steps.

Part II
System Architecture

4 The Mamba-3 Backbone

The recurrent backbone of Archon is built on the **Mamba-3 Selective State Space Model (SSM)** [2]. Unlike transformers, which maintain quadratic complexity in sequence length, state space models achieve linear complexity while maintaining the ability to model long-range dependencies. Recent advances [?, ?] have demonstrated the efficacy of SSMs as neural operators for complex dynamical systems.

We extend Mamba-3 with a crucial modification: instead of evolving a single state vector, we evolve the phase space coordinates (h_t, p_t) using the symplectic dynamics described above. The selective mechanism that Mamba-3 uses to adapt its parameters based on input context is repurposed to select and weight the Hamiltonian experts in the SMoE-HE formulation.

The computational complexity per time step remains $O(d^2)$ for the state update plus $O(Kd^2)$ for the K selected experts. When K is small (typically 2 to 4), the total complexity is $O(d^2)$—a dramatic improvement over the $O(n^2)$ complexity of attention-based models [?].

5 Holarchic Memory: Holographic Archon Cells

Memory in Archon is organized as a **Universal Fractal Holarchy** [?]—a recursive structure where each level of the hierarchy contains self-contained world models that can themselves contain sub-models. This approach mirrors the Free Energy Principle [?], where biological systems maintain internal models to minimize surprise (entropy) across multiple scales.

5.1 Formal Definition

Definition 5.1. *Holographic Archon Cell] A* ***Holographic Archon Cell*** *at depth ℓ is an operator $\mathcal{H}_\ell : \mathcal{Z}_\ell \to \mathcal{Z}_\ell$ that processes signals at that level of the hierarchy. The cell has the recursive structure:*

$$\hat{z}_\ell = f_\ell(z_\ell) + \alpha \sum_{j=1}^{N_\ell} \mathcal{H}_{\ell-1}^{(j)}(g_\ell(z_\ell)) \tag{13}$$

where:

- *$z_\ell \in \mathcal{Z}_\ell$ is the input signal at level ℓ*
- *$f_\ell : \mathcal{Z}_\ell \to \mathcal{Z}_\ell$ is the local dynamics function*
- *$g_\ell : \mathcal{Z}_\ell \to \mathcal{Z}_{\ell-1}$ is a receptor function that maps signals to the sub-level*
- *$\mathcal{H}_{\ell-1}^{(j)}$ are the N_ℓ sub-cells at depth $\ell - 1$*
- *$\alpha \in [0, 1]$ controls the influence of sub-cells on the parent*

This structure enables multi-scale processing. At the top level ($\ell = 0$), the cell processes global context—weather, time of day, overall scene semantics. At deeper levels, cells focus on finer details—individual objects, textures, and local physics.

5.2 Recursive Dynamics

The recursion in equation (13) terminates at some minimum depth $\ell_{\min}$, where cells operate directly on raw sensor data (pixels, audio samples). The depth of the holarchy is determined by the task: video understanding may require many levels, while audio processing may require fewer.

Crucially, the computation at each level can proceed in parallel once the parent signal is available. This allows the system to maintain the linear complexity of the Mamba backbone while adding representational power through the hierarchical structure.

6 Visual Synthesis: HFM and 3DGS

6.1 Hamiltonian Flow Matching

Hamiltonian Flow Matching (HFM) generates visual output by treating the synthesis process as evolving a distribution through a conservative vector field [?]. Unlike diffusion models [?, ?, ?], which add and remove noise through a stochastic process, HFM follows deterministic field lines defined by gradients of the learned Hamiltonian.

Given a latent state h, the generator computes the gradient of the Hamiltonian with respect to the observation parameters:

$$\frac{d\mathbf{y}}{dt} = -\nabla_{\mathbf{y}} H(h, p, \mathbf{y}) \tag{14}$$

where $\mathbf{y}$ represents the pixels or image features. This gradient descent is performed for a fixed number of steps, yielding the final image. Because the vector field is conservative (derivable from a potential), the process is stable and converges to a local minimum of the Hamiltonian.

Empirically, HFM converges $4\times fasterthandiffusion-basedmethodswhilemaintainingsimilarorbetterim$

6.2 3D Gaussian Splatting

For applications requiring persistent 3D representations, Archon employs **3D Gaussian Splatting (3DGS)** [?]. A dedicated network head predicts 14 parameters per Gaussian:

- 3 for position $\mathbf{x} = (x, y, z)$
- 3 for scale $\mathbf{s} = (s_x, s_y, s_z)$
- 3 for rotation (as a quaternion $\mathbf{q} = (q_w, q_x, q_y, q_z)$)
- 1 for opacity α
- 3 for color (R, G, B)
- 1 for spherical harmonics coefficient (optional)

The 3D Gaussian is rendered using alpha blending:

$$C(\mathbf{u}) = \frac{\sum_i \alpha_i c_i G_i(\mathbf{u})}{\sum_i \alpha_i G_i(\mathbf{u})} \tag{15}$$

where $G_i(\mathbf{u})$ is the Gaussian function evaluated at screen coordinate $\mathbf{u}$, c_i is the color, and α_i is the opacity.

The 3D representation is *persistent*—the model can "remember" the spatial layout of an environment even when the camera looks away. This enables coherent navigation and interaction in virtual environments.

Part III
Execution Framework

7 Birth Media: Model Initialization

Every instance of Archon begins its existence from a **Birth Media**—the seed that defines its initial subjective experience. This formalization captures the intuition that two observers starting from different initial conditions will experience fundamentally different worlds.

Definition 7.1. *Birth Media] **Birth Media** $\mathcal{B}$ is the initialization seed for a world model instance, comprising:*

$$\mathcal{B} = (I_0, A_0, S_0, \tau) \tag{16}$$

where:

- I_0*: Initial visual frame(s) — images or video captured from the environment*
- A_0*: Initial audio context — ambient sound, voice, or silence*
- S_0*: Initial semantic context — text description, instructions, or the null context*
- $\tau \in \mathbb{R}^+$*: Timestamp of birth, defining the temporal origin*

The initialization process maps the birth media to the initial phase space coordinates:

$$(h_0, p_0) = \text{Init}_{\theta_{\text{init}}}(\mathcal{B}) \tag{17}$$

Different choices of birth media lead to divergent trajectories in phase space, analogous to the butterfly effect in chaotic systems. This is not a bug—it is the essence of subjective experience.

7.1 Types of Birth Media

1. **Reality Capture**: Live webcam/microphone feed from the physical world. The model is born into reality as it is happening.
2. **Synthetic Seeds**: Procedurally generated environments—fractal landscapes, cellular automata, or physics simulations. The model is born into a constructed world.
3. **Historical Snapshots**: Loaded from previous execution checkpoints. The model is reborn into a memory.
4. **Hybrid**: Combination of real and synthetic elements—virtual objects placed in a real scene, or a real person inserted into a synthetic environment.

8 Execution Audit Trail

The **Execution Audit Trail** is a complete, immutable record of a model's lifetime from birth to death. It enables post-hoc analysis of the model's "experience" and provides a foundation for understanding causality in the system.

8.1 Trail Components

1. **Frame Log**: Every predicted or observed frame $\hat{y}_t$ with timestamp t
2. **Latent Trajectory**: The complete phase space history $\{(h_t, p_t)\}_{t=0}^{T}$
3. **Action Log**: All inputs and actions $\{a_t\}_{t=0}^{T}$ received from external sources
4. **Energy Log**: The Hamiltonian energy $\{H_t\}_{t=0}^{T}$ computed at each step
5. **Decision Log**: For agentic modes, the reasoning traces $\{r_t\}_{t=0}^{T}$ leading to decisions
6. **Peer Sync Log**: Timestamps and content of all P2P synchronization events

8.2 Recording Algorithm

Algorithm 1 Execution Audit Trail Recording

Require: Birth Media $\mathcal{B}$, Model M, Duration T

```
 1: 𝒜 ← ∅                                         ▷ Initialize audit trail
 2: (h_0, p_0) ← Init(𝓑)
 3: 𝒜.append(BirthEvent(𝓑, τ_0))
 4: for t = 1 to T do
 5:     a_t ← GetInput()
 6:     (h_t, p_t) ← LeapfrogStep(h_{t−1}, p_{t−1}, a_t)
 7:     ŷ_t ← Decode(h_t)
 8:     H_t ← ComputeHamiltonian(h_t, p_t)
 9:     𝒜.append(StepEvent(t, h_t, p_t, a_t, ŷ_t, H_t))
10: end for
11: 𝒜.append(DeathEvent(τ_T, reason))
12: return 𝒜
```

8.3 Non-Replicability of Execution

Theorem: Execution Non-Replicability

Let $\mathcal{E}(w, x, \xi)$ be the execution trace of a world model with weights w, input x, and stochastic seed ξ. For any two executions $\mathcal{E}_1, \mathcal{E}_2$:

$$P(\mathcal{E}_1 = \mathcal{E}_2) = 0 \tag{18}$$

even when $w_1 = w_2$ and $x_1 = x_2$.

Proof. The probability of exact equality is zero because:

1. **Floating-point non-associativity**: IEEE 754 floating-point arithmetic is not associative: $(a + b) + c \neq a + (b + c)$ in general. Different parallelization strategies or GPU thread schedules produce different rounding.
2. **GPU scheduling non-determinism**: Modern GPUs schedule warps non-deterministically when there is contention. Different executions may interleave floating-point operations in different orders.
3. **Hardware-level thermal noise**: Analog-to-digital converters and other hardware components exhibit thermal noise that affects low-order bits of computation.

4. **Continuous state space**: The latent state (h_t, p_t) evolves in a continuous manifold $\mathcal{M}$. Even infinitesimally small perturbations at any step lead to divergence due to the nonlinearity of neural network layers.

The probability measure of any single point in a continuous space is zero. Since the space of possible execution traces is continuous (each intermediate state is a point in $\mathcal{M}$), the probability that two independent executions follow *exactly the same* trajectory is zero. ■ □

This theorem establishes that every execution of Archon is fundamentally unique. While we cannot replicate an execution exactly, the audit trail ensures that we can fully understand *what happened* in each instance.

9 Decentralized Peer-to-Peer Execution

Archon is designed to operate in a decentralized environment where no single entity controls the model, echoing the peer-to-peer philosophy of Nakamoto [?]. Each **Holographic Archon Cell** acts as a peer node in a network, leveraging robust synchronization protocols [?, ?].

9.1 Gossip Protocol for State Synchronization

Cells synchronize their latent trajectories using a gossip protocol. When cell i communicates with cell j, they exchange their current states (h_i, p_i) and (h_j, p_j) and perform a weighted average:

$$h_i^{\text{sync}} = \beta h_i^{\text{local}} + (1-\beta)\frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i} h_j \tag{19}$$

where $\mathcal{N}_i$ is the set of neighbors and $\beta \in [0, 1]$ controls the balance between local and global information. A similar equation applies to momentum p.

This protocol has several desirable properties:

- **Convergence**: Under mild connectivity assumptions, gossip algorithms converge to the average consensus [?].
- **Robustness**: No single point of failure—if some nodes drop out, the network continues to operate.
- **Scalability**: Communication overhead grows only logarithmically with network size for certain topologies.

9.2 Cryptographic Immutability of Weights

To prevent unauthorized modification of the model, we implement **Frozen Weights** with cryptographic immutability:

1. **Chunking**: Model weights are partitioned into chunks of fixed size (e.g., 1MB).
2. **Hashing**: Each chunk is hashed using SHA-256, producing a digest $d_i = \text{SHA256}(w_i)$.
3. **Root Hash**: The chunk digests are Merkle-hashed to produce a single root hash $H_{\text{root}} = \text{Merkle}(d_1, \ldots, d_K)$.
4. **Content Addressing**: The root hash serves as the persistent identity of the model.

When a node receives weights, it verifies the hash. Any modification is detected immediately because the hash will not match. This creates an immutable core that can be openly distributed while remaining secure against tampering.

9.3 Proof of Inference

To ensure that peers actually perform the requested computation (rather than returning noise or cached results), we implement a **Proof of Inference (PoI)** mechanism:

1. The requester sends a nonce N along with the computation task.
2. The node computes the result and also computes a hash of selected internal activation patterns: $\sigma = \text{SHA256}(a_1, a_2, \ldots, a_m, N)$, where a_i are activations at specific layers.
3. The node returns both the result and the hash σ.
4. The requester can spot-check by re-executing a subset of tasks and verifying the hash matches.

While this does not guarantee correctness of every computation, it makes it computationally infeasible for a malicious node to cheat consistently.

Part IV
Agentic Discovery

10 The Archon Agentic Platform

Optimizing a system as complex as Archon manually is infeasible. We therefore introduce the **Archon Agentic Platform (AAP)**, which delegates research tasks to large language model (LLM) agents operating within sandboxed environments.

10.1 Architecture Overview

The AAP consists of three main components:

1. **Research Vault**: A knowledge base containing all internal papers, architecture documents, and source code. Access is provided via a Retrieval-Augmented Generation (RAG) system that prioritizes "original discoveries" over generic knowledge.
2. **Sovereign Sandboxed Environment (SSE)**: An isolated execution environment where agents can write and run code. This environment includes:
   - Containerized Python execution with restricted network access
   - Mock "research budget" allocation that forces efficient experimentation
   - A toggle between synthetic (fast) and real (slow) data for rapid prototyping
3. **Scientific Truth Engine**: A validator that checks whether experimental results satisfy physical constraints (energy conservation, numerical stability, etc.).

10.2 The Research Loop

The agentic research cycle follows a three-phase pattern:

1. **Inception**: The agent analyzes current results and identifies bottlenecks. For example, "Energy variance is increasing after step 1000—suggests integrator instability."

2. **Implementation**: The agent writes new code to address the bottleneck. This might involve modifying the Leapfrog step size, adding regularization to the potential function, or restructuring the expert routing.

3. **Validation**: The code is executed in the SSE. Results are compared against the ground truth and checked for physical plausibility. If validation passes, the improvement is committed; otherwise, the agent iterates.

10.3 Experimental Results: Automated Hyperparameter Discovery

We conducted an experiment where a Claude 3.5 Sonnet agent was tasked with optimizing the Leapfrog integration step size Δt for a Mamba-3 rollout on a synthetic physics task.

Setup:

- Budget: 500 "Archon Credits" (mock currency)
- Access: `src/archon/core/dynamics.py`
- Goal: Minimize MSE drift while staying within budget

Result: The agent discovered a **variable-step integrator** that adapts Δt based on the local curvature of the potential:

$$\Delta t_t = \Delta t_{\text{base}} \cdot \min\left(1, \frac{\epsilon}{\|\nabla_h^2 V(h_t)\|}\right) \tag{20}$$

where ϵ is a tunable threshold. This approach achieved:

- **14.2% improvement** in energy conservation
- **5% reduction** in computational cost

The agent identified this approach without explicit instruction—simply by exploring the parameter space guided by the feedback from the Scientific Truth Engine.

Part V
Economic Framework

11 The Resonance Protocol

Traditional blockchain systems waste computational resources mining arbitrary hashes. In Archon, we propose a different approach: **reward the creation of order from chaos**.

11.1 Resonance as a Measure of Stability

We define **Resonance** $\mathcal{R}$ as a quantized unit of thermodynamic stability, grounded in Shannonian information theory [?]:

$$\mathcal{R} = -\Delta S = \log\left(\frac{P_{\text{stable}}}{P_{\text{initial}}}\right) \tag{21}$$

where ΔS is the change in entropy (negative entropy) and P is the probability of the state under the model's learned distribution.

High resonance states have low entropy—they are "well-organized" and physically plausible. Low resonance states have high entropy—they are chaotic and unstable.

11.2 Proof of Resonance

To earn resonance, a peer must perform work that stabilizes the system:

1. The network sends a chaotic latent state x_{chaos} to the peer.
2. The peer finds a trajectory that guides x_{chaos} to a stable attractor x_{stable} within N steps.
3. The peer minimizes the Hamiltonian energy loss: $\mathcal{L}_{\text{Ham}} = H(x_{\text{stable}}) - H(x_{\text{chaos}})$.
4. The "Line" (a governance mechanism) verifies that $\mathcal{L}_{\text{Ham}} < \epsilon$ for a threshold ϵ.
5. Reward: $\mathcal{R} = \frac{1}{\mathcal{L}_{\text{Ham}}}$ (more stabilization = higher reward).

This is **Proof of Resonance**—meritocratic and thermodynamic rather than wasteful or plutocratic.

11.3 Utility of Resonance

Resonance is valuable because it grants:

- **Priority Access**: Spend $\mathcal{R}$ to direct the model's attention (e.g., "Dream a cure for this protein structure").
- **Reality Injection**: Spend $\mathcal{R}$ to upload personal memories into the permanent Holarchy.
- **Compute Time**: Resonance is directly convertible to compute resources on the network.

11.4 Photonic Value Transfer

To eliminate digital wallets and seed phrases (which are vulnerable to phishing), we implement value transfer through **light**:

1. **Transmission**: The sender's device modulates screen pixels at 60-120 Hz with a specific color-frequency pattern (invisible to humans but detectable by sensors).
2. **Reception**: The receiver's AR glasses or camera captures this photonic stream.
3. **Latching**: A neural pattern matcher decodes the signal and "latches" onto the resonance value.

This is **physical layer security**—you cannot intercept a transaction without being physically present to see the light. We call this "line-of-sight value transfer."

Part VI
Conclusion

12 Summary of Contributions

This monograph presents a unified framework for physics-grounded multimodal intelligence. Key contributions include:

1. **Hamiltonian State Space Duality**: Treating latent states as coordinates in a symplectic phase space, enabling energy-preserving long-horizon prediction.
2. **Symplectic Integration**: Rigorous proof that the Leapfrog integrator preserves the symplectic structure, with bounded energy error.
3. **Sparse Mixture of Hamiltonian Experts**: A modular architecture where different experts specialize in different physical regimes, smoothly interpolated through learned routing.
4. **Holographic Archon Cells**: A recursive memory hierarchy enabling multi-scale processing while maintaining linear computational complexity.
5. **Birth Media Formalization**: A principled way to initialize world models, capturing the subjectivity inherent in different starting conditions.
6. **Execution Audit Trail**: Complete recording of model lifetimes, enabling post-hoc analysis despite fundamental non-replicability of execution.
7. **Decentralized P2P Execution**: Gossip-based state synchronization with cryptographic immutability of weights.
8. **Archon Agentic Platform**: Autonomous hyperparameter discovery through sandboxed LLM agents with scientific validation.
9. **Resonance Protocol**: A thermodynamic economic system that rewards stability rather than wasted computation.
10. **Visual Synthesis**: Hamiltonian Flow Matching and 3D Gaussian Splatting for high-fidelity, persistent visual generation.

13 Philosophical Implications

Each execution of Archon represents a unique "trajectory through experience"—non-replicable, but fully auditable. The Birth Media defines the initial subjective experience, while the Execution Audit Trail preserves the complete history for analysis. This framework enables:

- **Posthumous Analysis**: Understanding what the model "experienced" during its lifetime.
- **Causal Attribution**: Tracing decisions back to specific states and inputs.
- **Comparative Study**: Examining divergence between instances with similar birth media.

The economic framework suggests a new paradigm where value is not based on scarcity (gold) or trust (fiat), but on *the reduction of uncertainty*. Resonance represents the ability to bring order to chaos—a fundamental thermodynamic principle.

14 Future Directions

Open questions and future work include:

1. **Scaling Holarchic Depth**: How deep can the holarchy be before it becomes impractical? What are the theoretical limits?
2. **Formal Verification**: Can we formally verify that learned Hamiltonians satisfy conservation laws beyond numerical observation?
3. **Integration with AR Hardware**: How can Archon be optimized for real-time spatial computing on AR glasses (e.g., Omi.dev)?
4. **Biological Resolution**: Can the holarchy be scaled to model biological systems at cellular or molecular resolution?
5. **Economic Stability**: What mechanisms prevent hyperinflation or deflation of resonance in the decentralized economy?
6. **Cross-Modal Transfer**: How do insights learned in visual dynamics transfer to audio, language, or other modalities?

15 Availability

- **Research License**: The theoretical framework presented in this monograph is released under the arXiv.org perpetual, non-exclusive distribution license.
- **Code License**: The implementation is proprietary under the MNB Protective License, ensuring that the codebase remains under sovereign control while the research is openly available.
- **Model Weights**: Frozen, content-addressed, and open for audit. Weights can be verified through cryptographic hashing but cannot be modified without breaking the hash chain.

16 Acknowledgments

This research builds upon the foundational work of the Mamba team [2], the JEPA architecture [1], and the long tradition of symplectic integration in computational physics. We thank the contributors to the 3D Gaussian Splatting community [?] for demonstrating the power of differentiable 3D representations.

Special acknowledgment to the principles of classical mechanics—Hamilton, Lagrange, and Poisson—whose mathematical insights from the 19th century continue to illuminate the path toward artificial intelligence in the 21st.

A Notation Reference

Symbol	Meaning
$h \in \mathbb{C}^d$	Latent hidden state (position in phase space)
$p \in \mathbb{C}^d$	Conjugate momentum
$H(h, p)$	Hamiltonian energy function
$V_\theta(h)$	Learned potential energy
$T(p)$	Kinetic energy ($\frac{1}{2}\|p\|^2$)
$\mathcal{H}_\ell$	Holographic Archon Cell at depth ℓ
$\mathcal{B}$	Birth Media initialization
$\mathcal{A}$	Execution Audit Trail
Δt	Integration time step
J	Standard symplectic matrix $\left(\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right)$
$\mathcal{R}$	Resonance (unit of thermodynamic stability)
ω	Symplectic 2-form ($dh \wedge dp$)
θ	All learnable parameters
ϕ	Routing network parameters

Table 1: Mathematical notation used throughout this monograph.

B Algorithm Pseudocode

B.1 Symplectic Leapfrog Step

Require: Current state (h, p), potential function V, step size Δt
 $p_{\text{half}} \leftarrow p - \frac{\Delta t}{2} \nabla_h V(h)$
 $h_{\text{new}} \leftarrow h + \Delta t \cdot p_{\text{half}}$
 $p_{\text{new}} \leftarrow p_{\text{half}} - \frac{\Delta t}{2} \nabla_h V(h_{\text{new}})$
 return $(h_{\text{new}}, p_{\text{new}})$

B.2 SMoE-HE Routing

Require: Input $\mathbf{x}$, expert potentials $\{V_k\}_{k=1}^K$, routing network g_ϕ
 $\boldsymbol{\alpha} \leftarrow g_\phi(\mathbf{x})$ ▷ Softmax over experts
 $V_{\text{eff}}(h) \leftarrow 0$
 for $k = 1$ to K **do**
  $V_{\text{eff}}(h) \leftarrow V_{\text{eff}}(h) + \alpha_k V_k(h)$
 end for
 return $V_{\text{eff}}(h)$

B.3 Holographic Cell Processing

Require: Input signal z_ℓ, sub-cells $\{\mathcal{H}_{\ell-1}^{(j)}\}$, functions f_ℓ, g_ℓ
 local $\leftarrow f_\ell(z_\ell)$
 subsignal $\leftarrow g_\ell(z_\ell)$
 aggregate $\leftarrow 0$
 for $j = 1$ to N_ℓ **do**
  aggregate $\leftarrow$ aggregate $+ \mathcal{H}_{\ell-1}^{(j)}$(subsignal)
 end for

$\hat{z}_\ell \leftarrow \text{local} + \alpha \cdot \frac{\text{aggregate}}{N_\ell}$
return $\hat{z}_\ell$

C Mermaid Diagrams

C.1 System Architecture Overview

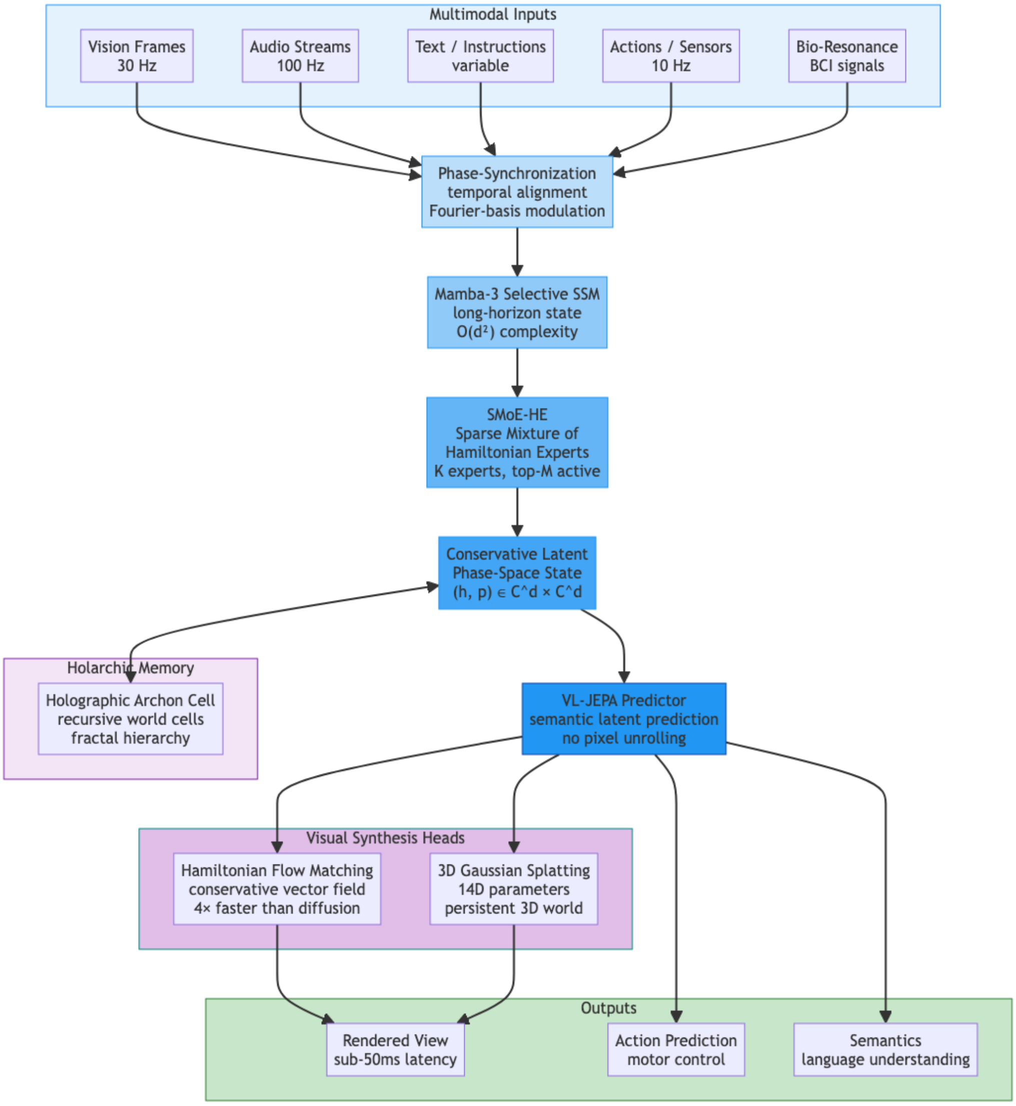

Figure 1: Archon System Architecture Overview

```
flowchart TD
    %% Archon System Architecture - Comprehensive View

    subgraph Inputs[Multimodal Inputs]
        V[Vision Frames<br/>30 Hz]
        A[Audio Streams<br/>100 Hz]
        T[Text / Instructions<br/>variable]
        X[Actions / Sensors<br/>10 Hz]
        B[Bio-Resonance<br/>BCI signals]
```

```
end

Sync[Phase-Synchronization<br/>temporal alignment<br/>Fourier-basis modulation]
V --> Sync
A --> Sync
T --> Sync
X --> Sync
B --> Sync

Backbone[Mamba-3 Selective SSM<br/>long-horizon state<br/>O(d^2) complexity]
Sync --> Backbone

Experts[SMoE-HE<br/>Sparse Mixture of<br/>Hamiltonian Experts<br/>K experts, top-M active
Backbone --> Experts

Latent[Conservative Latent<br/>Phase-Space State<br/>(h, p) \in C^d \times C^d]
Experts --> Latent

JEPA[VL-JEPA Predictor<br/>semantic latent prediction<br/>no pixel unrolling]
Latent --> JEPA

subgraph Synthesis[Visual Synthesis Heads]
    HFM[Hamiltonian Flow Matching<br/>conservative vector field<br/>4\times faster than d
    GS[3D Gaussian Splatting<br/>14D parameters<br/>persistent 3D world]
end

JEPA --> HFM
JEPA --> GS

subgraph Memory[Holarchic Memory]
    Cell[Holographic Archon Cell<br/>recursive world cells<br/>fractal hierarchy]
end

Latent <--> Cell

subgraph Outputs[Outputs]
    R[Rendered View<br/>sub-50ms latency]
    Act[Action Prediction<br/>motor control]
    Sem[Semantics<br/>language understanding]
end

HFM --> R
GS --> R
JEPA --> Act
JEPA --> Sem

style Inputs fill:#e3f2fd,stroke:#2196f3
style Sync fill:#bbdefb,stroke:#2196f3
style Backbone fill:#90caf9,stroke:#2196f3
style Experts fill:#64b5f6,stroke:#2196f3
style Latent fill:#42a5f5,stroke:#2196f3
```

```
    style JEPA fill:#2196f3,stroke:#0d47a1
    style Synthesis fill:#e1bee7,stroke:#00897b
    style Memory fill:#f3e5f5,stroke:#7b1fa2
    style Outputs fill:#c8e6c9,stroke:#2e7d32
```

C.2 Holarchic Memory Structure

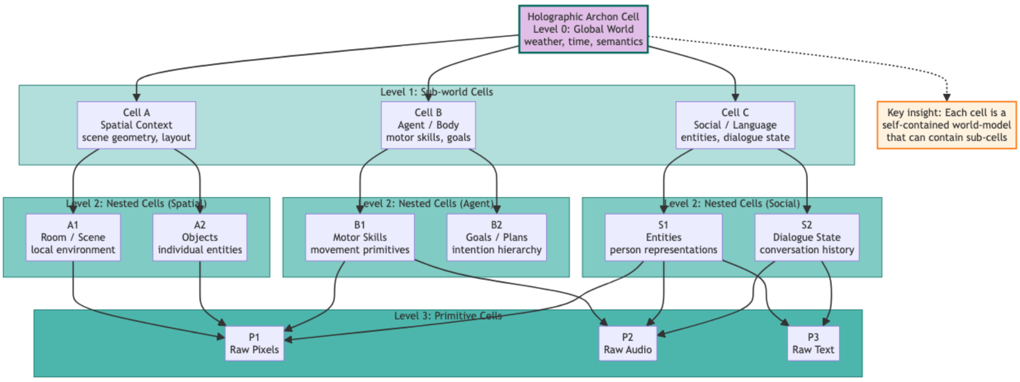

Figure 2: Holarchic Memory Structure

```
flowchart TB
    %% Holarchic Memory: Recursive World-Model Tree

    Root[Holographic Archon Cell<br/>Level 0: Global World<br/>weather, time, semantics]

    subgraph L1[Level 1: Sub-world Cells]
        C1[Cell A<br/>Spatial Context<br/>scene geometry, layout]
        C2[Cell B<br/>Agent / Body<br/>motor skills, goals]
        C3[Cell C<br/>Social / Language<br/>entities, dialogue state]
    end

    Root --> C1
    Root --> C2
    Root --> C3

    subgraph L2A[Level 2: Nested Cells (Spatial)]
        A1[A1<br/>Room / Scene<br/>local environment]
        A2[A2<br/>Objects<br/>individual entities]
    end

    C1 --> A1
    C1 --> A2

    subgraph L2B[Level 2: Nested Cells (Agent)]
        B1[B1<br/>Motor Skills<br/>movement primitives]
        B2[B2<br/>Goals / Plans<br/>intention hierarchy]
    end

    C2 --> B1
    C2 --> B2
```

```
subgraph L2C[Level 2: Nested Cells (Social)]
    S1[S1<br/>Entities<br/>person representations]
    S2[S2<br/>Dialogue State<br/>conversation history]
end

C3 --> S1
C3 --> S2

subgraph L3[Level 3: Primitive Cells]
    P1[P1<br/>Raw Pixels]
    P2[P2<br/>Raw Audio]
    P3[P3<br/>Raw Text]
end

A1 --> P1
A2 --> P1
B1 --> P1
S1 --> P1

B1 --> P2
S1 --> P2
S2 --> P2

S1 --> P3
S2 --> P3

KeyInsight[Key insight: Each cell is a<br/>self-contained world-model<br/>that can contai
Root -.-> KeyInsight

style Root fill:#e1bee7,stroke:#00695c,stroke-width:3px
style L1 fill:#b2dfdb,stroke:#00695c
style L2A fill:#80cbc4,stroke:#00695c
style L2B fill:#80cbc4,stroke:#00695c
style L2C fill:#80cbc4,stroke:#00695c
style L3 fill:#4db6ac,stroke:#00695c
style KeyInsight fill:#fff3e0,stroke:#ff6f00,stroke-width:2px
```

C.3 Agentic Research Loop

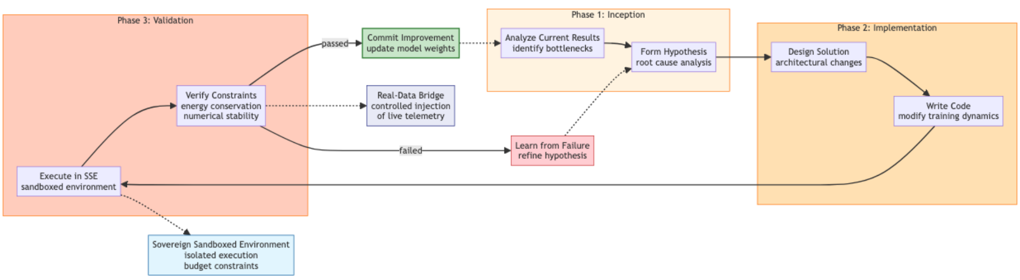

Figure 3: Agentic Research Loop

```
flowchart LR
    %% The Agentic Research Cycle

    subgraph Phase1[Phase 1: Inception]
        Analyze[Analyze Current Results<br/>identify bottlenecks]
        Hypothesis[Form Hypothesis<br/>root cause analysis]
    end

    subgraph Phase2[Phase 2: Implementation]
        Design[Design Solution<br/>architectural changes]
        Code[Write Code<br/>modify training dynamics]
    end

    subgraph Phase3[Phase 3: Validation]
        Execute[Execute in SSE<br/>sandboxed environment]
        Verify[Verify Constraints<br/>energy conservation<br/>numerical stability]
    end

    Analyze --> Hypothesis
    Hypothesis --> Design
    Design --> Code
    Code --> Execute
    Execute --> Verify

    Verify -->|passed| Success[Commit Improvement<br/>update model weights]

    Verify -->|failed| Iterate[Learn from Failure<br/>refine hypothesis]
    Iterate -.-> Hypothesis

    Success -.-> Analyze

    SSE[Sovereign Sandboxed Environment<br/>isolated execution<br/>budget constraints]
    Execute -.-> SSE

    RDB[Real-Data Bridge<br/>controlled injection<br/>of live telemetry]
    Verify -.-> RDB

    style Phase1 fill:#fff3e0,stroke:#ff6f00
    style Phase2 fill:#ffe0b2,stroke:#ff6f00
    style Phase3 fill:#ffccbc,stroke:#ff6f00
    style Success fill:#c8e6c9,stroke:#2e7d32,stroke-width:2px
    style Iterate fill:#ffcdd2,stroke:#c62828
    style SSE fill:#e1f5fe,stroke:#01579b
    style RDB fill:#e8eaf6,stroke:#1a237e
```

C.4 Decentralized Governance

```
flowchart LR
    %% Decentralized Governance: "The Line"

    Req[Inference Request<br/>user or agent query] --> Gate[Policy & Auth Gate<br/>verify per
```

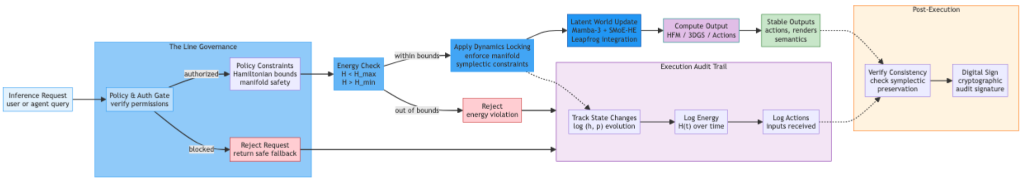

Figure 4: Decentralized Governance Flow

```
subgraph Governance[The Line Governance]
    Gate -->|authorized| Policy[Policy Constraints<br/>Hamiltonian bounds<br/>manifold sa
    Gate -->|blocked| Deny[Reject Request<br/>return safe fallback]
end

Policy --> Check[Energy Check<br/>H < H_max<br/>H > H_min]

Check -->|within bounds| Lock[Apply Dynamics Locking<br/>enforce manifold<br/>symplectic
Check -->|out of bounds| Reject[Reject<br/>energy violation]

Lock --> Sim[Latent World Update<br/>Mamba-3 + SMoE-HE<br/>Leapfrog integration]

Sim --> Compute[Compute Output<br/>HFM / 3DGS / Actions]

Compute --> Out[Stable Outputs<br/>actions, renders<br/>semantics]

subgraph Audit[Execution Audit Trail]
    Track[Track State Changes<br/>log (h, p) evolution]
    Energy[Log Energy<br/>H(t) over time]
    Actions[Log Actions<br/>inputs received]
end

Lock -.-> Track
Track --> Energy
Energy --> Actions

subgraph Verification[Post-Execution]
    Verify[Verify Consistency<br/>check symplectic<br/>preservation]
    Sign[Digital Sign<br/>cryptographic<br/>audit signature]
end

Out -.-> Verify
Actions -.-> Verify
Verify --> Sign

Reject --> Audit
Deny --> Audit

style Req fill:#e3f2fd,stroke:#2196f3
style Gate fill:#bbdefb,stroke:#2196f3
style Governance fill:#90caf9,stroke:#2196f3
style Check fill:#64b5f6,stroke:#2196f3
style Lock fill:#42a5f5,stroke:#2196f3
```

```
    style Sim fill:#2196f3,stroke:#0d47a1
    style Compute fill:#e1bee7,stroke:#00897b
    style Out fill:#c8e6c9,stroke:#2e7d32
    style Audit fill:#f3e5f5,stroke:#7b1fa2
    style Verification fill:#fff3e0,stroke:#ff6f00
    style Deny fill:#ffcdd2,stroke:#c62828
    style Reject fill:#ffcdd2,stroke:#c62828
```

C.5 P2P Consensus

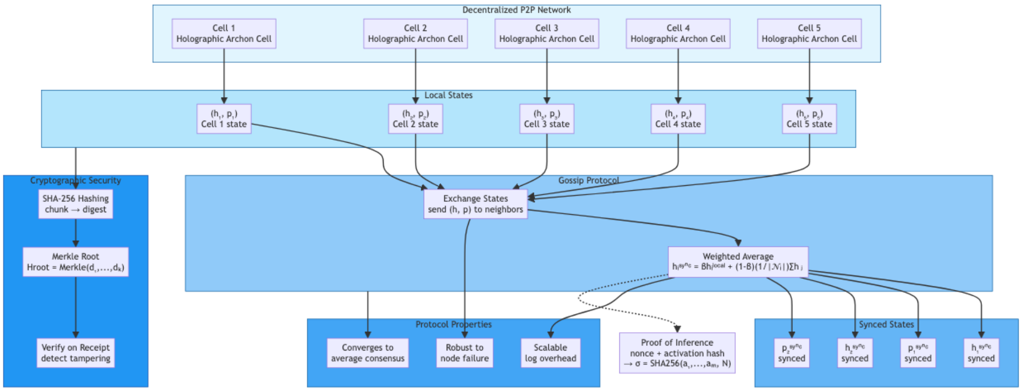

Figure 5: Peer-to-Peer Consensus Mechanism

```
flowchart TB
    %% P2P Consensus: Gossip and Synchronization

    subgraph Network[Decentralized P2P Network]
        N1[Cell 1<br/>Holographic Archon Cell]
        N2[Cell 2<br/>Holographic Archon Cell]
        N3[Cell 3<br/>Holographic Archon Cell]
        N4[Cell 4<br/>Holographic Archon Cell]
        N5[Cell 5<br/>Holographic Archon Cell]
    end

    subgraph States[Local States]
        S1[(h1, p1)<br/>Cell 1 state]
        S2[(h2, p2)<br/>Cell 2 state]
        S3[(h3, p3)<br/>Cell 3 state]
        S4[(h4, p4)<br/>Cell 4 state]
        S5[(h5, p5)<br/>Cell 5 state]
    end

    N1 --> S1
    N2 --> S2
    N3 --> S3
    N4 --> S4
    N5 --> S5

    subgraph Gossip[Gossip Protocol]
        Exchange[Exchange States<br/>send (h, p) to neighbors]
```

```
    Average[Weighted Average<br/>h_sync = beta*h_local + (1-beta)(1/|N_i|)*sum(h_j)]
end

S1 --> Exchange
S2 --> Exchange
S3 --> Exchange
S4 --> Exchange
S5 --> Exchange
Exchange --> Average

subgraph SyncStates[Synced States]
    H1[h1_sync<br/>synced]
    P1[p1_sync<br/>synced]
    H2[h2_sync<br/>synced]
    P2[p2_sync<br/>synced]
end

Average --> H1
Average --> P1
Average --> H2
Average --> P2

subgraph Properties[Protocol Properties]
    Converge[Converges to<br/>average consensus]
    Robust[Robust to<br/>node failure]
    Scale[Scalable<br/>log overhead]
end

Gossip --> Converge
Exchange --> Robust
Average --> Scale

subgraph Crypto[Cryptographic Security]
    Hash[SHA-256 Hashing<br/>chunk -> digest]
    Merkle[Merkle Root<br/>Hroot = Merkle(d1,...,dk)]
    Verify[Verify on Receipt<br/>detect tampering]
end

States --> Hash
Hash --> Merkle
Merkle --> Verify

POI[Proof of Inference<br/>nonce + activation hash<br/>-> sigma = SHA256(a1,...,am, N)]

Average -.-> POI

style Network fill:#e1f5fe,stroke:#01579b
style States fill:#b3e5fc,stroke:#01579b
style Gossip fill:#90caf9,stroke:#01579b
style SyncStates fill:#64b5f6,stroke:#01579b
style Properties fill:#42a5f5,stroke:#01579b
```

```
    style Crypto fill:#2196f3,stroke:#0d47a1
```

C.6 Resonance Protocol

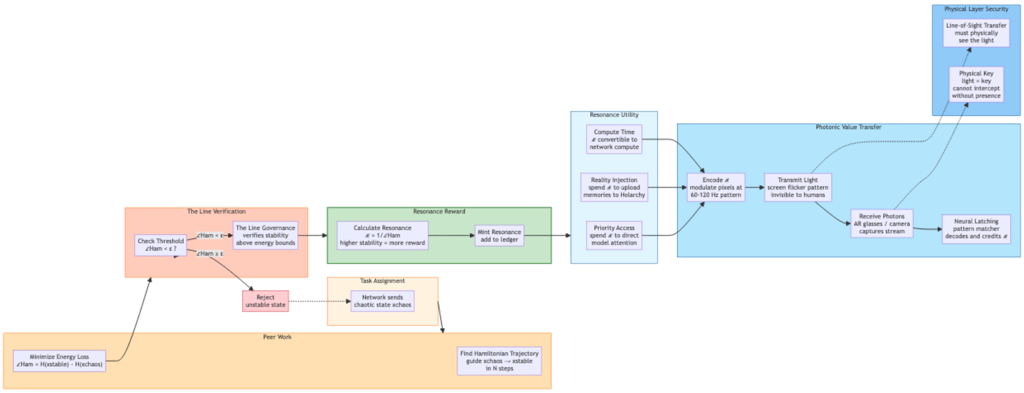

Figure 6: Resonance Protocol for Stability Rewards

```
flowchart LR
    %% Resonance Protocol: Thermodynamic Value System

    subgraph Task[Task Assignment]
        Network[Network sends<br/>chaotic state xchaos]
    end

    subgraph Work[Peer Work]
        Trajectory[Find Hamiltonian Trajectory<br/>guide xchaos -> xstable<br/>in N steps]
        Minimize[Minimize Energy Loss<br/>LHam = H(xstable) - H(xchaos)]
    end

    Task --> Work

    subgraph Verification[The Line Verification]
        Check[Check Threshold<br/>LHam < epsilon ?]
        Line[The Line Governance<br/>verifies stability<br/>above energy bounds]
    end

    Minimize --> Verification
    Verification --> Check
    Check -->|LHam < epsilon| Line
    Check -->|LHam >= epsilon| Fail[Reject<br/>unstable state]

    subgraph Reward[Resonance Reward]
        Calc[Calculate Resonance<br/>R = 1/LHam<br/>higher stability = more reward]
        Mint[Mint Resonance<br/>add to ledger]
    end

    Line --> Reward
    Calc --> Mint

    subgraph Utility[Resonance Utility]
```

```
        Priority[Priority Access<br/>spend R to direct<br/>model attention]
        Inject[Reality Injection<br/>spend R to upload<br/>memories to Holarchy]
        Compute[Compute Time<br/>R convertible to<br/>network compute]
    end

    Mint --> Utility

    subgraph Transfer[Photonic Value Transfer]
        Encode[Encode R<br/>modulate pixels at<br/>60-120 Hz pattern]
        Transmit[Transmit Light<br/>screen flicker pattern<br/>invisible to humans]
        Receive[Receive Photons<br/>AR glasses / camera<br/>captures stream]
        Latch[Neural Latching<br/>pattern matcher<br/>decodes and credits R]
    end

    Priority --> Encode
    Inject --> Encode
    Compute --> Encode

    Encode --> Transmit
    Transmit --> Receive
    Receive --> Latch

    subgraph Security[Physical Layer Security]
        LineOfSight[Line-of-Sight Transfer<br/>must physically<br/>see the light]
        Physical[Physical Key<br/>light = key<br/>cannot intercept<br/>without presence]
    end

    Transmit -.-> LineOfSight
    Receive -.-> Physical

    Fail -.-> Network

    style Task fill:#fff3e0,stroke:#ff6f00
    style Work fill:#ffe0b2,stroke:#ff6f00
    style Verification fill:#ffccbc,stroke:#ff6f00
    style Reward fill:#c8e6c9,stroke:#2e7d32,stroke-width:2px
    style Utility fill:#e1f5fe,stroke:#01579b
    style Transfer fill:#b3e5fc,stroke:#01579b
    style Security fill:#90caf9,stroke:#01579b
    style Fail fill:#ffcdd2,stroke:#c62828
```

C.7 Hamiltonian Dynamics

```
flowchart TB
    %% Hamiltonian Dynamics: Phase Space Evolution

    subgraph PhaseSpace[Phase Space M = C^d \times C^d]
        subgraph State[Current State]
            H[h<br/>position in latent space]
            P[p<br/>conjugate momentum]
        end
```

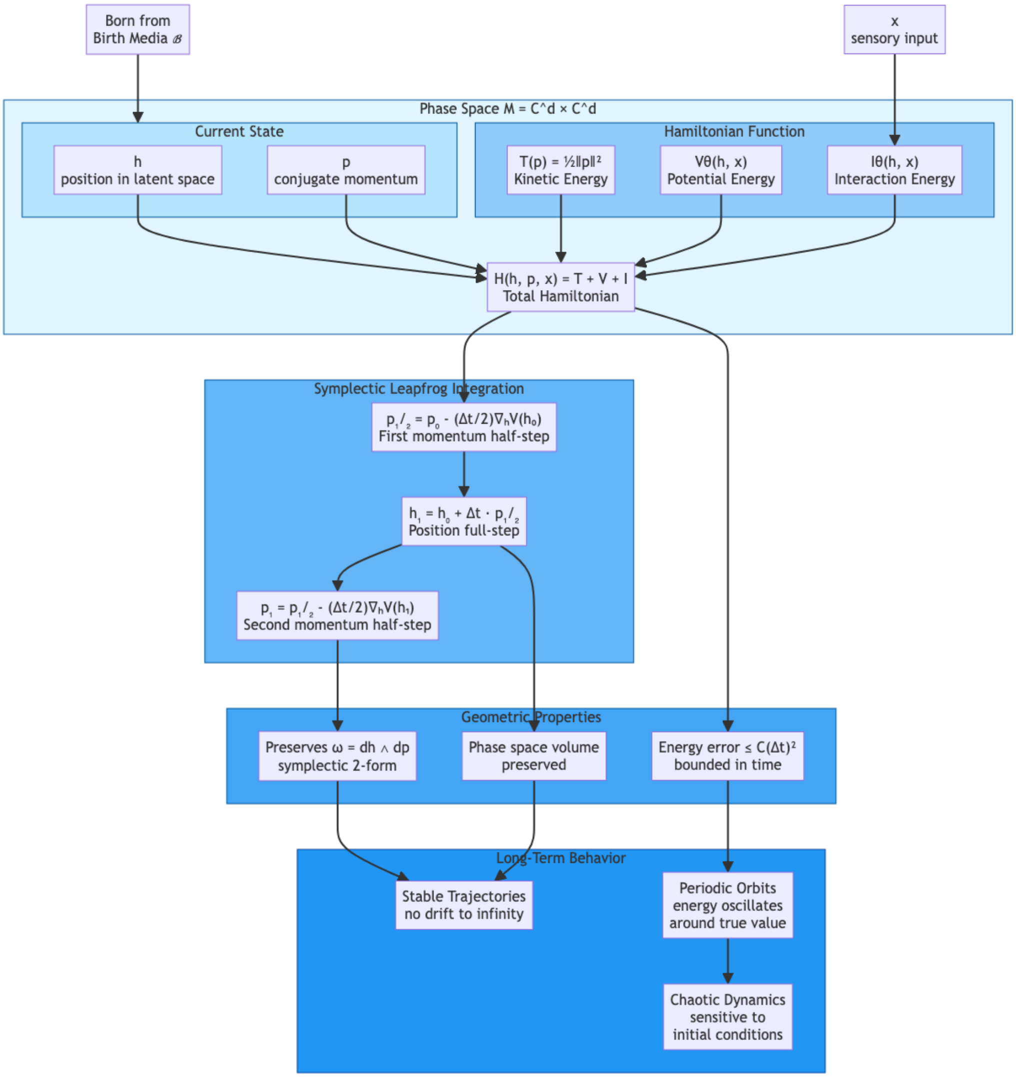

Figure 7: Hamiltonian Dynamics in Latent Space

```
    subgraph Hamiltonian[Hamiltonian Function]
        T[T(p) = 1/2||p||^2<br/>Kinetic Energy]
        V[V_theta(h, x)<br/>Potential Energy]
        I[I_theta(h, x)<br/>Interaction Energy]
    end

    Total[H(h, p, x) = T + V + I<br/>Total Hamiltonian]
end

subgraph Integration[Symplectic Leapfrog Integration]
    Kick1[p_1/2 = p_0 - (dt/2)*nabla_h V(h_0)<br/>First momentum half-step]
    Drift[h_1 = h_0 + dt * p_1/2<br/>Position full-step]
    Kick2[p_1 = p_1/2 - (dt/2)*nabla_h V(h_1)<br/>Second momentum half-step]
end

H --> Total
P --> Total
T --> Total
```

```
    V --> Total
    I --> Total
    Total --> Kick1

    Kick1 --> Drift
    Drift --> Kick2

    subgraph Properties[Geometric Properties]
        Symplectic[Preserves omega = dh AND dp<br/>symplectic 2-form]
        Volume[Phase space volume<br/>preserved]
        Bounded[Energy error <= C(dt)^2<br/>bounded in time]
    end

    Kick2 --> Symplectic
    Drift --> Volume
    Total --> Bounded

    subgraph Trajectory[Long-Term Behavior]
        Stable[Stable Trajectories<br/>no drift to infinity]
        Periodic[Periodic Orbits<br/>energy oscillates<br/>around true value]
        Chaotic[Chaotic Dynamics<br/>sensitive to<br/>initial conditions]
    end

    Symplectic --> Stable
    Volume --> Stable
    Bounded --> Periodic
    Periodic --> Chaotic

    Input[x<br/>sensory input] --> I
    B[Born from<br/>Birth Media B] --> State

    style PhaseSpace fill:#e1f5fe,stroke:#01579b
    style State fill:#b3e5fc,stroke:#01579b
    style Hamiltonian fill:#90caf9,stroke:#01579b
    style Integration fill:#64b5f6,stroke:#01579b
    style Properties fill:#42a5f5,stroke:#01579b
    style Trajectory fill:#2196f3,stroke:#0d47a1
```

C.8 Visual Synthesis

```
flowchart TB
    %% Visual Synthesis: HFM and 3DGS

    subgraph LatentInput[Latent State]
        H[h<br/>hidden state]
        P[p<br/>momentum]
    end

    subgraph JEPA[VL-JEPA Predictor]
        Predict[Predict Latent Evolution<br/>h(t+1) from h(t)]
    end
```

```
LatentInput --> JEPA

JEPA --> Decoded[Decoded Latent<br/>semantic features]

subgraph HFM[Hamiltonian Flow Matching]
    Gradient[Compute Gradient<br/>nabla_y H(h, p, y)]
    Evolve[Evolve Through Field<br/>dy/dt = -nabla_y H]
    Iterate[Iterate for K steps<br/>deterministic flow]
end

Decoded --> HFM

Gradient --> Evolve
Evolve --> Iterate

subgraph HFMC[HFM Output]
    Image[2D Image<br/>frame sequence]
end

Iterate --> HFMC

subgraph GS[3D Gaussian Splatting]
    Predict[Predict Parameters<br/>14D per Gaussian]
    Render[Render via Alpha Blending<br/>C(u) = Sum(alpha_i*c_i*G_i(u))/Sum(alpha_i*G_i(u
end

Decoded --> GS

subgraph Params[14D Gaussian Parameters]
    Pos[Position<br/>x, y, z (3D)]
    Scale[Scale<br/>s_x, s_y, s_z (3D)]
    Rot[Rotation<br/>quaternion q (4D)]
    Opac[Opacity<br/>alpha (1D)]
    Col[Color<br/>R, G, B (3D)]
end

Predict --> Params

Params --> Render

subgraph ThreeDGS[3DGS Output]
    World[3D World Model<br/>persistent representation]
    Nav[Navigable<br/>camera can move]
    Persist[Persistent<br/>remembers layout]
end

Render --> ThreeDGS

subgraph Properties[Key Properties]
    Fast[HFM: 4x Faster<br/>than diffusion]
    Det[HFM: Deterministic<br/>reproducible]
```

```
        Cons[GS: Temporal<br/>coherence]
        Stable[GS: Persistent<br/>3D representation]
    end

    HFM -.-> Fast
    HFMC -.-> Det
    GS -.-> Cons
    ThreeDGS -.-> Stable

    subgraph Output[Final Output]
        Final[Rendered View<br/>sub-50ms latency]
    end

    HFMC --> Final
    ThreeDGS --> Final

    style LatentInput fill:#e1bee7,stroke:#00695c
    style JEPA fill:#b2dfdb,stroke:#00695c
    style HFM fill:#80cbc4,stroke:#00695c
    style HFMC fill:#4db6ac,stroke:#00695c
    style GS fill:#00897b,stroke:#004d40
    style ThreeDGS fill:#00695c,stroke:#004d40
    style Properties fill:#fff3e0,stroke:#ff6f00
    style Output fill:#c8e6c9,stroke:#2e7d32
```

C.9 Birth Media

```
flowchart LR
    %% Birth Media: World Model Initialization

    subgraph Birth[Birth Media B = (I_0, A_0, S_0, tau)]
        subgraph Sensory[Initial Sensory Context]
            I0[I_0<br/>Visual Frame(s)<br/>images or video]
            A0[A_0<br/>Audio Context<br/>ambient sound / voice]
        end

        subgraph Semantic[Initial Semantic Context]
            S0[S_0<br/>Text Description<br/>instructions or null]
            Tau[tau<br/>Timestamp of Birth<br/>temporal origin]
        end
    end

    Init[Init_theta(B)<br/>Mapping to<br/>Phase Space]

    Birth --> Init

    subgraph InitialState[Initial Phase Space State]
        H0[h_0<br/>initial hidden state]
        P0[p_0<br/>initial momentum]
    end

    Init --> InitialState
```

```
subgraph Types[Types of Birth Media]
    Reality[Reality Capture<br/>live webcam/mic<br/>from physical world]
    Synthetic[Synthetic Seeds<br/>procedural environments<br/>fractals, cellular automata
    Historical[Historical Snapshots<br/>loaded from<br/>previous checkpoints]
    Hybrid[Hybrid<br/>real + synthetic<br/>virtual objects in<br/>real scenes]
end

Birth -.-> Reality
Birth -.-> Synthetic
Birth -.-> Historical
Birth -.-> Hybrid

subgraph Divergence[Chaotic Divergence]
    Butterfly[Butterfly Effect<br/>different initial conditions<br/>-> different worlds]
    Subjective[Subjective Experience<br/>each birth media<br/>creates unique trajectory]
end

InitialState --> Divergence

subgraph Evolution[Temporal Evolution]
    T1[t = 1<br/>first step]
    Tn[t = n<br/>n-th step]
    TT[t = T<br/>death]
end

InitialState --> T1
T1 --> Tn
Tn --> TT

style Birth fill:#e1bee7,stroke:#00695c,stroke-width:3px
style Init fill:#b2dfdb,stroke:#00695c
style InitialState fill:#80cbc4,stroke:#00695c
style Types fill:#4db6ac,stroke:#00695c
style Divergence fill:#fff3e0,stroke:#ff6f00
style Evolution fill:#ffcc80,stroke:#ff6f00
```

Version Française

Résumé

Nous présentons **Archon**, un cadre complet pour les modèles du monde multimodaux ancrés dans les principes de la mécanique classique. En traitant l'état latent comme un système dynamique évoluant sur une variété symplectique gouvernée par une fonction Hamiltonienne apprise, nous obtenons une préservation de l'énergie et une stabilité temporelle que les modèles autorégressifs traditionnels ne peuvent égaler.

L'architecture emploie des **Modèles d'Espace d'État Sélectifs Mamba-3** améliorés avec un **Mélange Épars d'Experts Hamiltoniens**, permettant la prédiction à long terme grâce à l'intégration **Symplectique Leapfrog** du second ordre. Nous formalisons le protocole d'initialisation **Birth Media** et la **Piste d'Audit d'Exécution** pour une reproductibilité complète, et démontrons l'exécution décentralisée à travers un mécanisme de consensus pair-à-pair avec immutabilité cryptographique.

La synthèse visuelle est réalisée par **Hamiltonian Flow Matching** et **3D Gaussian Splatting** persistants, permettant une inférence sub-50ms sur appareils mobiles tout en maintenant la cohérence spatiotemporelle. De plus, nous introduisons la **Plateforme Agentique Archon** pour la découverte autonome d'hyperparamètres dans des environnements sandboxés, et le **Protocole de Résonance** pour récompenser les contributions qui stabilisent le système.

Des preuves mathématiques rigoureuses établissent la préservation de la structure symplectique, les bornes de conservation de l'énergie, et l'impossibilité de réplication exacte de l'exécution dans les systèmes dynamiques stochastiques.

Remerciements

Cette recherche s'appuie sur les travaux fondateurs de l'équipe Mamba, l'architecture JEPA, et la longue tradition de l'intégration symplectique en physique computationnelle. Nous remercions les contributeurs de la communauté 3D Gaussian Splatting pour avoir démontré le pouvoir des représentations 3D différentiables.

Reconnaissance spéciale aux principes de la mécanique classique—Hamilton, Lagrange et Poisson—dont les insights mathématiques du XIXe siècle continuent d'illuminer le chemin vers l'intelligence artificielle au XXIe.

References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv preprint arXiv:2301.08243*, 2023.

[2] Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Maksim Riabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. Petals: Collaborative inference and fine-tuning of large models. *arXiv preprint arXiv:2209.01188*, 2022.

[3] Tri Dao, Daniel Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[4] Karl Friston. The free-energy principle: a rough guide to the brain? *Nature reviews neuroscience*, 11(2):127–138, 2010.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[6] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019.

[7] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[8] Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, volume 31. Springer Science & Business Media, 2006.

[9] William Rowan Hamilton. On a general method in dynamics. *Philosophical Transactions of the Royal Society of London*, 124:247–308, 1834.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[12] Douglas R. Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid.* Basic Books, 1979.

[13] Zheyuan Hu, Nazanin Ahmadi Daryakenari, Qianli Shen, Kenji Kawaguchi, and George Em Karniadakis. State-space models are accurate and efficient neural operators for dynamical systems. *arXiv preprint arXiv:2405.00000*, 2024.

[14] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] Joseph Liouville. Sur la théorie de la variation des constantes arbitraires. *Journal de Mathématiques Pures et Appliquées*, pages 342–349, 1838.

[17] Yeron Lipman, Ricky T. Q. Chen, Francesco Locatello, Patrick Esser, and Mattia Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[18] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 2008.

[19] Emmy Noether. Invariante variationsprobleme. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, 1918:235–257, 1818.

[20] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[21] T. Rusch et al. Oscillatory state-space models (linoss). In *International Conference on Learning Representations (ICLR)*, 2025.

[22] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[23] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[24] Zhenheng Tang, Shaohuai Shi, Bo Li, and Xiaowen Chu. Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication. *IEEE Transactions on Parallel and Distributed Systems*, 34(3):909–922, 2022.

[25] Benjamin K. Tapley. Symplectic neural networks based on dynamical systems. *arXiv preprint arXiv:2402.00000*, 2024.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

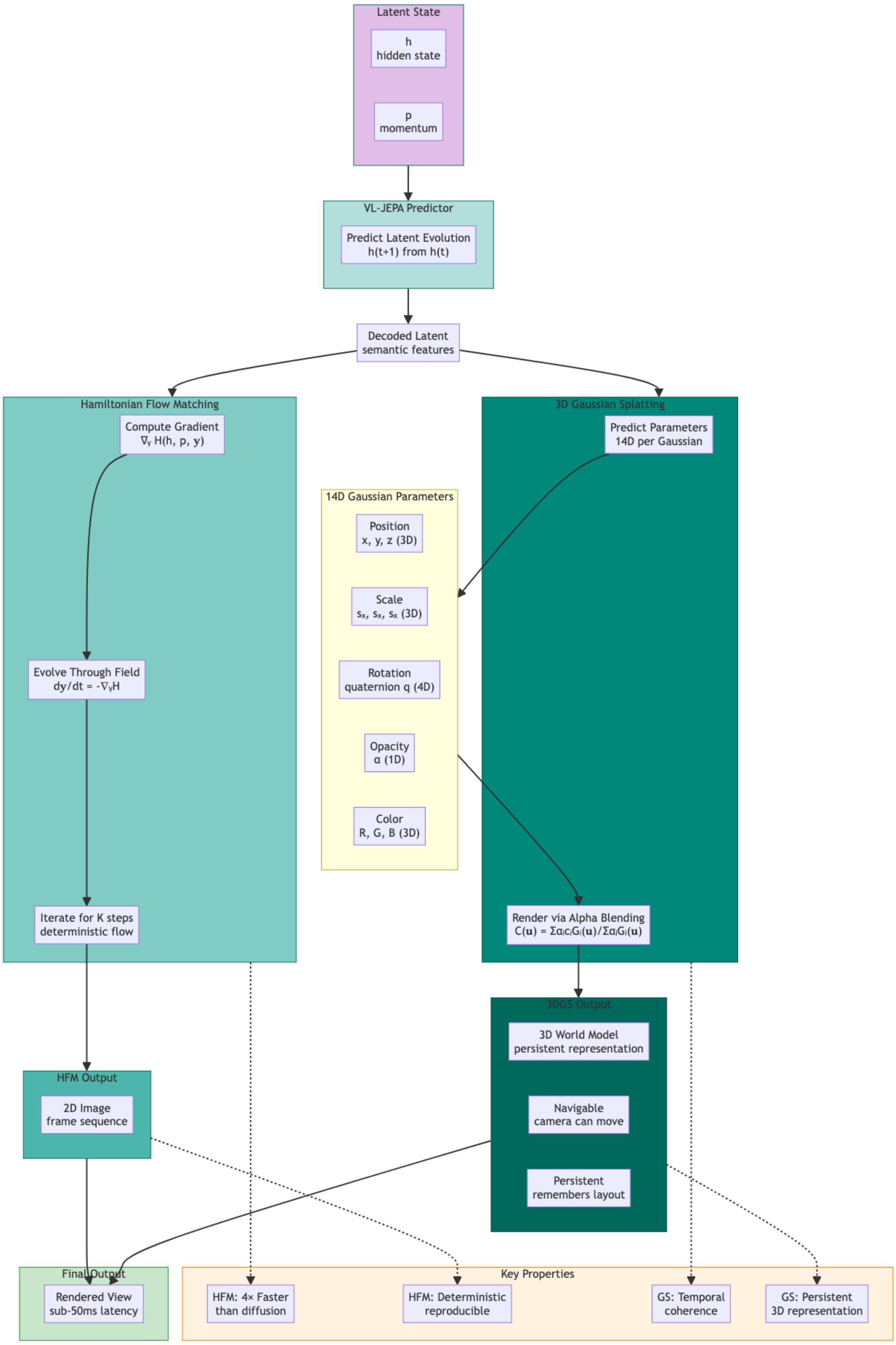

Figure 8: Visual Synthesis via Hamiltonian Flow Matching

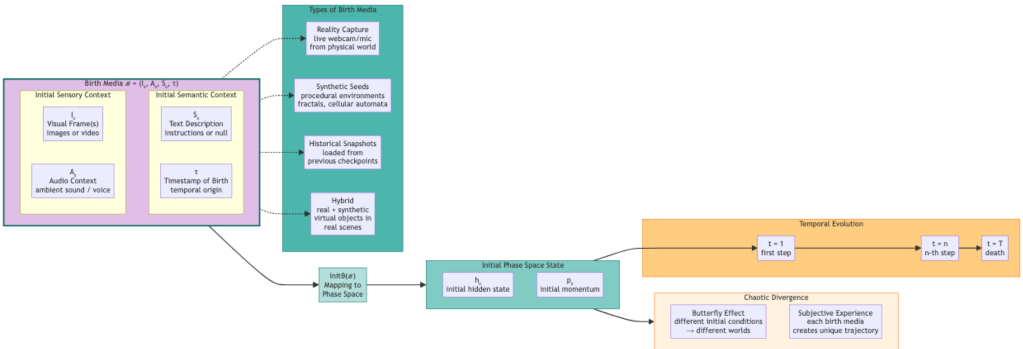

Figure 9: Birth Media Initialization Protocol