Learning When to Act:

Interval-Aware Reinforcement Learning with Predictive Temporal Structure

Davide Di Gioia
University College London
ucsigi@ucl.ac.uk

March 2026

Abstract

Autonomous agents operating in continuous environments must decide not only *what* to do, but *when* to act. We introduce a lightweight adaptive temporal control system that learns the optimal interval between cognitive ticks from experience, replacing ad hoc biologically-inspired timers with a principled learned policy. The policy state is augmented with a novel *predictive hyperbolic spread* signal (a "curvature signal" shorthand) derived from hyperbolic geometry: the mean pairwise Poincaré distance among n sampled futures embedded in the Poincaré ball $\mathbb{B}_c^n$. High spread indicates a branching, uncertain future and drives the agent to act sooner; low spread signals predictability and permits longer rest intervals. We further propose an *interval-aware reward* that explicitly penalises inefficiency relative to the chosen wait time, correcting a systematic credit-assignment failure of naive outcome-based rewards in timing problems. We additionally introduce a *joint spatio-temporal embedding* (**ATCPG-ST**) that concatenates independently normalised state and position projections in the Poincaré ball; spatial trajectory divergence provides an independent timing signal unavailable to the state-only variant (**ATCPG-SO**). This extension raises mean hyperbolic spread (κ) from 1.88 to 3.37 and yields a further +5.8% efficiency gain over the state-only baseline. Ablation experiments across five random seeds demonstrate that (i) learning is the dominant efficiency factor (+54.8% over no-learning), (ii) hyperbolic curvature provides significant complementary gain (+26.2% over geometry-free control), (iii) the combined system achieves +22.8% efficiency over the fixed-interval baseline, and (iv) adding spatial position information to the curvature embedding yields an additional +5.8%.

Contents

1 Introduction

The rapid capability expansion of Large Language Models (LLMs) has catalysed the deployment of autonomous agents and multi-agent systems (MAS) in open-ended, continuous environments. In these settings, an agent's efficacy is bounded not only by *what* action it takes, but *when* it chooses to act. As agents transition from isolated chatbots to persistent digital workers, the temporal cadence of their reasoning cycles directly dictates their computational overhead, responsiveness, and ultimate task success.

Despite sophisticated advances in task delegation and multi-agent routing, the internal temporal pacing of individual agents remains remarkably primitive. Most contemporary orchestration frameworks rely on either strictly reactive, event-driven triggers (an agent only wakes when messaged) or hand-tuned, fixed-interval polling loops (e.g., `sleep(N)` calls). Bio-mimetic heuristic formulas (such as hard-coded "fatigue multipliers" or "ultradian rhythms") offer a veneer of adaptability, but remain brittle: they encode the system designer's static prior rather than the agent's lived experience, and they categorically fail to adapt as task complexity and environmental volatility shift.

To address this structural blind spot, we propose **Adaptive Temporal Control via Predictive Geometry (ATCPG)**, a lightweight, fully autonomous pacing system that allows an agent to learn its optimal cognitive interval dynamically. ATCPG shifts temporal control from a fixed network orchestrator to an internal, learned policy driven by the agent's own epistemic uncertainty. The framework is constructed from four interacting components:

1. **Learned pacing policy**: a linear associative bandit that continuously updates the wait interval after every cognitive tick via reward-weighted regression.
2. **Predictive hyperbolic spread** (informally, a "curvature signal"), a geometric signal capturing the dispersion of n predicted future states in the Poincaré ball $\mathbb{B}_c^n$. The metric's natural boundary expansion aggressively amplifies diverging trajectories, signalling the agent to act sooner.
3. **Interval-aware shaping reward**: a dynamic learning signal that prevents the catastrophic credit-assignment failure common to standard RL timing problems by explicitly pricing the chosen interval.
4. **Joint spatio-temporal embedding (ATCPG-ST)**: an extension that augments the policy state with Poincaré-projected spatial position vectors. Spatial trajectory divergence is an independent timing signal: branching navigation trees fan out in position space before belief space, and the Poincaré ball's boundary-expansion property amplifies both signals in a unified geometric language. ATCPG-ST raises mean hyperbolic spread (κ) by 1.79× and efficiency by +5.8% over the state-only baseline (**ATCPG-SO**); it degrades gracefully to **ATCPG-SO** whenever position data are absent.

The paper makes the following contributions:

- We formalise the *temporal pacing problem* as an RL problem over a continuous action space (Section 3).
- We derive the *interval-aware reward* and prove it avoids the credit-assignment failure of naive outcome reward in the pacing setting (Section 5).

- We introduce *predictive hyperbolic spread* (a "curvature signal" shorthand) as a timing signal, grounded in the Poincaré ball model of hyperbolic geometry (Section 6).
- We propose **ATCPG-ST**, a *joint spatio-temporal embedding* (Section 6.5) that lifts position trajectories into the same Poincaré ball as state embeddings, motivated by the observation that spatial trajectory divergence is an independent leading indicator of decision uncertainty unavailable to the state-only ATCPG-SO. We demonstrate empirically that the combined signal raises curvature by 1.79× and efficiency by +5.8% on a controlled ablation, and that the ordering $\eta_{\mathrm{SO}} < \eta_{\mathrm{ST}}$ replicates on a live GPT-4.1 deployment (Section 7.9).
- We provide systematic ablation evidence that each component contributes independently to efficiency (Section 7).

2 Related Work

To situate ATCPG within the broader landscape of artificial intelligence, we contextualise our framework across four distinct domains: multi-agent orchestration, temporal abstraction in reinforcement learning, geometric representation, and autonomous cognitive loops.

Decentralised orchestration and emergent synchronisation. A growing body of work in multi-agent systems (MAS) shifts temporal coordination away from centralised schedulers toward decentralised, self-organising mechanisms. In contemporary LLM-agent frameworks, execution is increasingly modelled as event-driven computation on structured graphs, where agents are triggered by dependency resolution, message arrivals, or verification outcomes rather than a global clock [Yang et al., 2025, Nalagatla, 2025]. In parallel, control-theoretic and multi-agent reinforcement learning literatures study how global temporal regularities emerge from local interaction rules, including consensus-style coordination and communication-efficient protocols that yield synchronised collective behaviour without explicit central timing [Oh et al., 2025, Stoorvogel et al., 2025, Erofeeva et al., 2025].

Crucially, these approaches resolve temporal structure primarily at the *network layer*: they formalise *when agents should interact with one another*, given topological dependencies and local communication signals. The internal deliberation cadence of an individual agent is often treated as a reactive black box (an agent "runs" when invoked), rather than as a decision variable with its own uncertainty sensitivity, credit assignment, and temporal cost. Chen et al. [2025] identify this absence of intrinsic self-monitoring as a core unsolved deficiency in AI autonomy. ATCPG targets this orthogonal gap by formalising intrinsic cognitive pacing at the node level: a continuously running agent learns how long to wait between self-initiated cognitive ticks. This intrinsic pacing is complementary to decentralised orchestration rather than a replacement: agents may adapt their own tick intervals while still exhibiting emergent group rhythm via the oscillator and Kuramoto-style phase coupling [Kuramoto, 1984] detailed in Section 4.4.

Temporal abstraction in RL. Options [Sutton et al., 1999] and macro-actions allow agents to choose the *duration* of a commitment, but they operate over discrete intervals and do not adapt the base decision frequency. Semi-MDPs [Puterman, 1994] generalise to continuous sojourn times but require a full transition model; ATCPG operates in the model-free, online contextual-bandit setting. Oudeyer and Kaplan [2007] motivate self-paced learning from progress signals. LIDA [Franklin et al., 2007] uses a fixed cognitive cycle; we replace the fixed cycle with a learned one. Circadian and

ultradian-inspired architectures [Wang and Aamodt, 2015] hard-code period parameters that we instead learn.

Reward shaping for timing. Ng et al. [1999] study potential-based shaping; we propose a different class of shaping that explicitly accounts for temporal cost. Dewey [2014] addresses utility indifference, but does not consider interval selection as the decision variable. Neither specifically resolves the temporal credit-assignment failure we identify in pacing, necessitating our derivation of an interval-aware reward.

Hyperbolic representation and uncertainty. Nickel and Kiela [2017] demonstrate that hierarchical structure embeds more faithfully in $\mathbb{B}_c^n$ than in Euclidean space due to exponential volume growth near the boundary. Ganea et al. [2018] extend neural networks to hyperbolic space. We use the Poincaré ball not for representation learning but as a *geometric divergence measure* between predicted futures, quantifying epistemic conflict to drive a temporal control policy, a novel application.

Structural blind spots in agent orchestration. Di Gioia [2026] identify a pervasive architectural blind spot in modern multi-agent orchestration stacks: their native scheduling layers route execution blindly without geometric perception of how failures propagate through the graph. They demonstrate that geometry-aware sidecars employing dynamic hyperbolic metric switching close this observability gap. ATCPG takes direct inspiration from this critique: where Di Gioia apply hyperbolic geometry to solve the *routing* (spatial) blind spot in multi-agent execution graphs, we apply it to solve the *pacing* (temporal) blind spot in continuous autonomous loops.

Autonomous cognitive loops and self-directed timing. The *Agentic Heartbeat Pattern* [Mendonca, 2025] addresses *who* coordinates with whom across an organisational hierarchy but does not formalise when a single agent should self-initiate. The SPOC system [Zhao et al., 2025] interleaves solution generation and verification but remains externally triggered; no runtime pacing adaptation occurs. ATCPG's daemon loop runs continuously between external requests with no external trigger.

Chen et al. [2025] identify "absence of intrinsic self-monitoring" and "absence of intrinsic agency" as core unsolved deficiencies of current AI systems, explicitly calling for the type of self-directed temporal control we realise here. To our knowledge, ATCPG is the first concrete, learnable temporal-control module addressing the self-directed timing layer they identify as absent.

InSeC [Li et al., 2024] bakes error-correction behaviour into model weights via negative-sample training. The result is a static inference policy activated by an external user query; no runtime pacing adaptation occurs. ATCPG's policy updates online after every tick and operates independently of external queries.

3 Problem Formulation

3.1 Temporal Pacing as a Contextual Bandit

Let an agent operate a cognitive loop with discrete ticks $t = 1, 2, \ldots$ At each tick the agent observes a state $s_t \in \mathbb{R}^d$ and selects an interval $\Delta t_t \in [\Delta t_{\min}, \Delta t_{\max}]$ before the next tick. After sleeping for Δt_t seconds the agent acts, observes wellbeing scalar $w_t \in [0, 1]$ (a composite health/success signal), and receives reward r_t. The agent's goal is to maximise the long-run average efficiency:

$$\mathcal{J}(\pi) \;=\; \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \frac{r_t}{\Delta t_t}. \tag{1}$$

This objective explicitly penalises using more time than necessary to achieve a unit of reward.

3.2 State Representation

The state vector $s_t \in \mathbb{R}^6$ fed to the policy contains:

$$s_t = \big[\; \underbrace{p_t}_{\text{priority}} ,\; \underbrace{f_t}_{\text{fatigue}} ,\; \underbrace{\Delta w_t}_{\text{wellbeing delta}} ,\; \underbrace{\rho_t}_{\text{performance}} ,\; \underbrace{\sin\phi_t}_{\text{oscillator phase}} ,\; \underbrace{\kappa_t}_{\text{hyperbolic spread}} \;\big]^\top \tag{2}$$

where ϕ_t is an internal phase variable that captures emergent rest/activity cycles (Section 4.4).

4 Learned Pacing Policy

4.1 Linear Policy

We parameterise the policy as a linear map:

$$\hat{\Delta t}(\boldsymbol{\theta}, s) \;=\; \theta_{\text{bias}} + \theta_p\, p + \theta_f\, f + \theta_w\, \Delta w + \theta_\rho\, \rho + \theta_\phi \sin\phi + \theta_\kappa\, \kappa, \tag{3}$$

with output clamped to $[\Delta t_{\min}, \Delta t_{\max}]$. The initial weights encode sensible priors: $\theta_p < 0$ (urgency shortens wait), $\theta_f > 0$ (fatigue lengthens wait), $\theta_\kappa < 0$ (uncertainty shortens wait).

4.2 Online Weight Update

After observing reward r_t we update the weight vector via the direct online rule:

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} + \alpha\, r_t \cdot s_t. \tag{4}$$

Interpretation of the update rule. The update (4) is *not* an unbiased policy-gradient estimator for the clipped policy (3). It is best understood as a lightweight online *reward-weighted linear regression* (equivalently, a linear associative bandit rule) that increases weights on features co-occurring with positive realised reward. This choice prioritises simplicity and low computational overhead (one vector update per tick) over unbiased gradient estimation.

For reference, the exact REINFORCE [Williams, 1992] step for a linear-Gaussian policy $\pi(\Delta t \mid s)$ would include the deviation term $(\Delta t_t - \hat{\Delta t}_t)$:

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} + \alpha\, r_t\, \frac{\Delta t_t - \hat{\Delta t}_t}{\sigma^2} \cdot s_t. \tag{5}$$

We omit that term, treating the post-tick reward r_t as an approximately stationary response to the tick context s_t over the adaptation time-scale of α. Empirically, the simplified rule provides a stable and effective online update for the pacing problem studied here.

4.3 State-Dependent Exploration

We inject multiplicative noise with probability ε_{eff}:

$$\varepsilon_{\text{eff}}(s_t) \;=\; \varepsilon_0 \cdot (1 - |\Delta w_t|), \tag{6}$$

so the agent explores freely when wellbeing is stable ($|\Delta w_t| \approx 0$) and exploits its current policy when wellbeing is volatile ($|\Delta w_t| \approx 1$). This reverses the conventional uncertainty → explore heuristic. It is not contradictory, but contextually adapted: in a continuous autonomous loop, high volatility ($|\Delta w_t| \approx 1$) signals acute risk or task failure, demanding immediate exploitation of robust, known intervals to regain stability. Conversely, stable wellbeing indicates the system has the safety margin required to experiment with pacing efficiency.

4.4 Internal Oscillator

The internal phase ϕ_t evolves as:

$$\phi_{t+1} = (\phi_t + \omega_t) \bmod 2\pi, \qquad \omega_{t+1} = \text{clip}\big(\omega_t + \alpha r_t \cdot 0.01,\ [0.001, 0.2]\big), \tag{7}$$

where ω_t is the phase velocity. Positive rewards accelerate the rhythm; negative rewards slow it. This produces emergent rest/activity cycles whose period is determined endogenously by task structure rather than by a hyperparameter.

Multi-agent synchronisation. When N clocks operate in parallel, Kuramoto coupling [Kuramoto, 1984] nudges phases toward the group mean:

$$\phi_t^{(i)} \;\leftarrow\; \phi_t^{(i)} + \lambda\Big(\bar{\phi}_t - \phi_t^{(i)}\Big), \quad \bar{\phi}_t = \frac{1}{N}\sum_j \phi_t^{(j)}, \tag{8}$$

producing collective rhythm without central coordination.

5 Interval-Aware Reward

5.1 The Credit-Assignment Failure of Outcome Reward

Proposition 1 (Reward-direction failure)**.** *Under the online linear update* (4) *with naive reward* $r_t = \Delta w_t$ *and feature* $f_t > 0$*, the update* $\theta_f \mathrel{+}= \alpha r_t f_t$ *decreases* θ_f *when* $r_t < 0$ *(overload), producing* shorter *intervals when the agent is fatigued, opposite to the desired behaviour.*

Proof. If the agent is overloaded, $\Delta w_t < 0 \Rightarrow r_t < 0$. The update $\theta_f \mathrel{+}= \alpha \cdot (\text{negative}) \cdot f_t$ decreases θ_f, reducing the positive contribution $\theta_f \cdot f_t$ to the interval output (3). □

This is a fundamental misalignment: the update ascribes the negative outcome to the fatigue feature and reduces the fatigue-correction behaviour.

5.2 Proposed Interval-Aware Reward

We decompose the reward into three additive components:

$$r_t \;=\; \underbrace{2\,\frac{\Delta w_t}{\Delta t_t}}_{\text{efficiency}} \;+\; \underbrace{1.5\,o_t\,\frac{\Delta t_t}{\Delta t_{\text{base}}}}_{\text{spacing bonus}} \;+\; \underbrace{1.0\,\frac{\kappa_t}{\Delta t_t}}_{\text{spread brake}}, \tag{9}$$

where $o_t = \max(0, -\Delta w_t)$ is the overload magnitude.

Relation to the evaluation objective. Equation (9) is a shaping signal used for the online update (4); it is *not* divided by Δt_t again as in (1). The $1/\Delta t_t$ factors in the efficiency and curvature-brake terms encourage the agent to favour short intervals, acting as a surrogate for the temporal cost in (1). This avoids double-counting: we optimise the shaping signal (9) directly and evaluate on the metric (18).

Efficiency term. $\Delta w_t/\Delta t_t$ measures wellbeing gained *per second of interval chosen*. This correctly penalises long idle periods that produce the same outcome as short ones, driving the agent toward tighter action packing when the environment is cooperative.

Spacing bonus. $o_t \cdot (\Delta t_t/\Delta t_{\text{base}})$ rewards long intervals *when the agent is overloaded*. Under update rule (4) this creates a positive gradient on θ_f, directly fixing the failure in Proposition 1.

Spread brake. $\kappa_t/\Delta t_t$ rewards small intervals when future uncertainty is high. This encodes the intuition that a branching future demands more frequent re-evaluation, formalised in Section 6.

6 Predictive Hyperbolic Spread via Hyperbolic Geometry

6.1 Poincaré Ball Primer

The Poincaré ball model $(\mathbb{B}_c^n, g^c)$ is the open unit ball $\{x \in \mathbb{R}^n : c\|x\|^2 < 1\}$ equipped with the Riemannian metric

$$g_x^c = \lambda_x^{c\,2}\, g^E, \qquad \lambda_x^c = \frac{2}{1 - c\|x\|^2}, \tag{10}$$

where $c > 0$ is the curvature parameter and g^E is the Euclidean metric. The geodesic distance between $x, y \in \mathbb{B}_c^n$ is:

$$d_{\mathbb{H}}(x, y) \;=\; \frac{2}{\sqrt{c}} \operatorname{arctanh}\Big(\sqrt{c}\,\|{-x} \oplus_c y\|\Big), \tag{11}$$

where $\oplus_c$ is the Möbius addition:

$$x \oplus_c y \;=\; \frac{(1 + 2c\langle x, y\rangle + c\|y\|^2)\, x \;+\; (1 - c\|x\|^2)\, y}{1 + 2c\langle x, y\rangle + c^2\|x\|^2\|y\|^2}. \tag{12}$$

A key property used below: equal-Euclidean-step displacements near the boundary of $\mathbb{B}_c^n$ produce exponentially larger geodesic distances than the same displacements near the origin. This means $d_{\mathbb{H}}$ *amplifies* divergence in future predictions, making small differences in predicted outcomes measurable.

6.2 Future Embedding

Given a world model $\mathcal{W}$, we sample n future state trajectories $\{z^{(i)}\}_{i=1}^n$ of horizon h. Each trajectory summary vector $z^{(i)} \in \mathbb{R}^d$ is embedded into $\mathbb{B}_c^m$ via:

$$\varphi(z) \;=\; \operatorname{proj}\Big(\sigma \cdot \frac{z'}{\|z'\|}\Big), \qquad z' = \text{pad/trim}(z, m), \tag{13}$$

where $\sigma \in (0,1)$ is a scale factor (default 0.9) and $\text{proj}(x) = x/\max(1, \|x\|/r_{\max})$ ensures the result lies inside $\mathbb{B}_c^n$. The zero vector is embedded to the origin.

6.3 Spread Estimator

Definition 1 (Predictive Hyperbolic Spread)**.** *Given n embedded futures $\{e^{(i)}\}_{i=1}^n \subset \mathbb{B}_c^n$, the* predictive hyperbolic spread *is defined as:*

$$\kappa = \underbrace{\mu_{ij}}_{\text{mean spread}} + \underbrace{\sigma_{ij}^2}_{\text{variance of spread}}, \tag{14}$$

where μ_{ij} and σ_{ij}^2 are the mean and variance of $\{d_{\mathbb{H}}(e^{(i)}, e^{(j)})\}_{i<j}$.

Terminology note. Although the manifold's sectional curvature is fixed by the parameter c (constant throughout $\mathbb{B}_c^n$), we use **"curvature signal"** (κ) as an evocative shorthand for this spread statistic. The name highlights that κ is *amplified* by the manifold's expanding geometry near the boundary, not that it measures the manifold curvature itself.

Remark 1. *Adding variance to mean penalises* heterogeneous *divergence, a mix of near-identical and wildly differing futures, which is more alarming than uniform spreading and deserves a stronger re-evaluation signal.*

Proposition 2 (Zero curvature for identical futures)**.** *If all n predicted futures are identical, $\kappa = 0$.*

Proof. $e^{(i)} = e^{(j)}$ for all i, j implies $d_{\mathbb{H}}(e^{(i)}, e^{(j)}) = 0$ (since $\| - x \oplus_c x\| = 0$ by Möbius addition), so $\mu_{ij} = \sigma_{ij}^2 = 0$. □

Geometric regime characterisation. The Poincaré metric amplifies pairwise distances selectively based on two geometric quantities: the *radial position* $r = \|e\|$ of the embeddings and the *angular divergence* between perturbed samples. The conformal factor $\lambda_c^2 = 4/(1-r^2)^2$ grows from ≈ 4 near the origin to ≈ 111 at $r = 0.9$, creating three qualitatively distinct curvature regimes.

Proposition 3 (Three-regime amplification)**.** *Let $u, v \in \mathbb{B}_c^n$ with $\|u\| = \|v\| = r$ and angle $\theta = \angle(u, v)$ between them. For small angular perturbations $\delta = 1 - \cos\theta \ll 1$, the squared Poincaré distance satisfies*

$$d_c(u,v)^2 \approx \frac{8\, r^2\, \delta}{(1-r^2)^2}. \tag{15}$$

Thus κ is jointly amplified by radial position through $(1-r^2)^{-2}$ and by angular spread through δ.

Proof. By definition, $d_c(u,v) = \text{arccosh}(1+X)$ where $X = 2\|u-v\|^2/(1-r^2)^2$ for $c = 1$ and equal radii. The law of cosines gives $\|u-v\|^2 = 2r^2(1-\cos\theta) = 2r^2\delta$, so $X = 4r^2\delta/(1-r^2)^2$. For small X, the Taylor expansion $\text{arccosh}(1+X) \approx \sqrt{2X}$ yields $d_c(u,v)^2 \approx 2X = 8r^2\delta/(1-r^2)^2$. □

Remark 2. *Equation (15) is a small-angle, equal-radius approximation. The implementation uses the exact* arccosh *formula $d_c(x,y) = \text{arccosh}(1 + 2c\|x-y\|^2/[(1-c\|x\|^2)(1-c\|y\|^2)])$ [Ungar, 2008], which handles arbitrary radii and is numerically stable.*

We demonstrated this amplification mechanism empirically under controlled construction, using MC-dropout [Gal and Ghahramani, 2016] as a surrogate for future-state spread. Table 1 reports the mean predictive curvature κ for three synthetically constructed state topologies under $N = 200$ dropout samples at rate $p = 0.20$.

Table 1: MC-dropout predictive curvature κ across three geometric regimes ($N = 200$ samples, $p = 0.20$, $d = 6$). The Poincaré boundary amplification selectively elevates conflicted states.

Regime	Construction	r	κ	$\kappa/\kappa_{\text{confident}}$
Conflicted uncertainty	Opposing dominant features at boundary	≈ 0.90	6.22	10.4×
Confident prediction	Stable signal, ball interior	≈ 0.37	0.60	1.0×
Uninformative noise	Near-zero state, Euclidean limit	≈ 0.12	0.21	0.35×

Both discriminability ratios massively exceed the baseline: Conflicted vs. Confident = 10.4×; Conflicted vs. Noise = 29.6×. This confirms that κ does not merely measure spread, it measures *directional conflict amplified by proximity to the ball boundary*, a property unique to hyperbolic geometry.

6.4 Integration with the Pacing Policy

The spread signal enters the policy (3) with weight $\theta_\kappa < 0$, so high spread decreases the predicted interval. In the reward (9) the spread brake term $\kappa/\Delta t_t$ creates a positive gradient on θ_κ when the interval is short, reinforcing the *speed-up under uncertainty* behaviour (act sooner when future spread is high).

6.5 Joint Spatio-Temporal Embedding

The state-only embedding (13) captures *what* the future will look like, the uncertainty over predicted *content*, but not *where* the agent or world will be spatially. Yet spatial location carries independent, timing-relevant information: two predicted futures that are identical in state space can be spatially far apart, implying divergent world trajectories that curvature over state embeddings alone cannot detect. In navigation, multi-step planning, and execution graphs, spatial trajectory divergence is often a *stronger* leading indicator of decision uncertainty than state spread: a branching navigation tree fans out in position space well before it fans out in belief space.

The Poincaré ball is an ideal joint container for both signals: its boundary-expansion property amplifies state uncertainty and spatial trajectory fan-out in the same geometric language, without requiring separate encoders or manual scale matching. We therefore extend ATCPG by augmenting the policy state with Poincaré-projected position vectors; we call the resulting system **ATCPG-ST** (*Spatio-Temporal*) to distinguish it from the state-only variant **ATCPG-SO**. ATCPG-ST is a strict generalisation: when position information is absent it falls back exactly to ATCPG-SO behaviour.

Formally, when the world model produces both predicted state vectors and predicted spatial position vectors $\{p^{(i)}\}_{i=1}^{n} \subset \mathbb{R}^k$, we form a *joint* spatio-temporal embedding by concatenating independently normalised Poincaré projections:

$$\psi^{(i)} \;=\; \text{proj}\Big(\big[\varphi(z^{(i)};\, m_s) \,\|\, \varphi(p^{(i)};\, m_p)\big]\Big), \tag{16}$$

where m_s and m_p are the state and position embedding dimensions (defaults 6 and 3), $\|$ denotes concatenation, and the outer proj re-projects the $(m_s + m_p)$-dimensional vector into $\mathbb{B}_c^{m_s+m_p}$. Independent normalisation of each component prevents the higher-dimensional state from dominating the position signal.

Predictive curvature is then computed exactly as in (14) but over the joint embeddings $\{\psi^{(i)}\}$. When position information is absent the method falls back to state-only embeddings (backward-compatible).

Spatial divergence amplifies κ. Because the Poincaré metric expands near the ball boundary, trajectories that diverge in position space ($\|p^{(i)} - p^{(j)}\|$ large) produce joint embeddings near the boundary, and their geodesic distances $d_{\mathbb{H}}(\psi^{(i)}, \psi^{(j)})$ are substantially larger than those computed from state embeddings alone. The empirical effect is a higher κ and correspondingly shorter intervals whenever the agent's predicted spatial trajectories fan out, a desirable signal in navigation or multi-step planning tasks where spatial divergence correlates with decision uncertainty (Section 7.8).

Proposition 4 (Spatial monotonicity, non-saturated regime)**.** *Let $z^{(i)} \in \mathbb{R}^{m_s}$ be fixed and let $p^{(i)} \in \mathbb{R}^k$ be position vectors. Define joint embeddings $\psi^{(i)}(\varepsilon)$ by replacing $p^{(i)}$ with $\varepsilon p^{(i)}$ in (16). Assume there exists $\varepsilon_{\max} > 0$ such that for all $\varepsilon \in [0, \varepsilon_{\max}]$ the outer proj$(\cdot)$ in (16) does not activate (no clipping). Then $\kappa^{\mathrm{ST}}(\varepsilon)$ is non-decreasing in ε on $[0, \varepsilon_{\max}]$. In particular, if $p^{(i)} \neq p^{(j)}$ for some $i \neq j$, then for sufficiently small $\varepsilon > 0$, $\kappa^{\mathrm{ST}}(\varepsilon) > \kappa^{\mathrm{SO}}$.*

Proof sketch. Without clipping, scaling $p^{(i)} \mapsto \varepsilon p^{(i)}$ increases pairwise Euclidean separations of the position components monotonically in ε. For embeddings bounded away from the boundary, the Poincaré distance is monotone in $\|x - y\|$ for fixed radii, so each pairwise $d_{\mathbb{H}}(\psi^{(i)}, \psi^{(j)})$ is non-decreasing. Their mean and variance are therefore non-decreasing, giving $\kappa^{\mathrm{ST}}(\varepsilon) \geq \kappa^{\mathrm{SO}}$. □

7 Experiments

7.1 Simulation Environment

We evaluate in a synthetic environment where each tick simulates a cognitive cycle with stochastic outcomes:

$$\begin{aligned} d_t &\sim \text{Gaussian}(d_{\text{base}}(o_t) - 30p_t,\ 20), \\ \Delta w_t &\sim \text{Gaussian}(\mu_w(o_t),\ 0.05), \\ \text{success}_t &= \mathbf{1}[\neg o_t \ \vee\ p_t > 0.7], \end{aligned} \tag{17}$$

where $d_{\text{base}} \in \{50, 200\}$ ms for non-overloaded and overloaded, $\mu_w \in \{+0.1, -0.2\}$, $o_t \sim$ Bernoulli(0.3). All agents use $\Delta t_{\min} = 10\,\text{s}$, $\Delta t_{\max} = 300\,\text{s}$, base interval 60 s.

7.2 Metrics

Efficiency (primary).

$$\eta = \frac{1}{T} \sum_{t=1}^{T} \frac{\text{success}_t}{\Delta t_t}. \tag{18}$$

Performance score. Mean success rate.

Wellbeing stability. Standard deviation of the wellbeing trajectory (lower is better).

7.3 Baselines and Ablations

We compare five configurations, plus a privileged reference baseline and a spatial extension (Table 2):

Full model (ATCPG) All components enabled; curvature inferred geometrically from predicted futures, no privileged information.

−Learning Weights frozen; fixed interval equals base.

−Curvature Curvature feature set to 0; policy learns from remaining features.

−Interval reward Simple outcome reward $r_t = 2\Delta w_t + 0.5/d_t$ replaces (9).

−Exploration $\varepsilon_0 = 0$ (deterministic policy).

ATCPG-ST (positions) Full model extended with joint spatio-temporal embedding (16); positions are independently drawn 3-D vectors correlated with overload state.

TC (privileged)† `TemporalController` with curvature computed from a Gaussian conditioned on the directly observed overload flag o_t. This constitutes an *upper bound*: the best achievable efficiency when a clean oracle overload signal is available.

Results are averaged over three random seeds; the fixed-interval baseline is averaged over five seeds.

7.4 Results

Learning dominates. Removing learning halves efficiency (−54.8%). No other component comes close. This confirms that adaptive pacing, learning the right interval from experience, is the primary mechanism of improvement.

Hyperbolic spread provides significant gain. The −26.2% drop in the −Curvature condition demonstrates that the geometric future-spread signal contributes substantially beyond what the non-geometric features (priority, fatigue, wellbeing, performance) alone can capture. The hyperbolic spread signal provides a *prospective* signal: it encodes what is *about to happen* rather than what *just happened*, giving the policy information unavailable to non-geometric policy learners.

Interval-aware reward corrects credit assignment. The −15.1% drop in the −Interval reward condition empirically validates Proposition 1: the naive outcome reward produces a systematic bias against slowing down under overload. Replacing it with (9) removes this bias and improves efficiency.

Exploration stabilises. The modest −3.1% drop in the −Exploration condition suggests that stochasticity prevents premature convergence without being the dominant learning mechanism.

Comparison to fixed-interval baseline. On the primary metric, the full model achieves +22.8% improvement over the fixed-interval baseline across five seeds ($\eta = 0.0162$ vs. 0.0132), with consistent advantage on performance score (0.80 vs. 0.79). Average reward is near-equal (0.4996 vs. 0.5009); we regard efficiency (18) as the correct primary metric since it explicitly accounts for temporal cost.

Privileged upper bound (TC). `TemporalController` achieves $\eta = 0.0275$ when granted direct access to the overload flag o_t to construct its curvature estimate. This is −5.2% below the full ATCPG model (0.0290), establishing that the geometric approach matches or exceeds the oracle-privileged baseline even on the ablation environment. Because TC directly observes o_t while ATCPG does not, this result constitutes a *lower bound* on ATCPG's advantage in realistic partially-observable settings (see the full 500-tick head-to-head analysis in Section 7.7 where the gap widens to +72.5%).

Table 2: Efficiency scores η (18) for the full ATCPG model, ablated variants (averaged over 3 seeds), the fixed-interval baseline (averaged over 5 seeds), and `TemporalController` as a privileged upper bound. Δ is the relative change versus the full model.

Variant	**Efficiency η**	**Δ vs Full**	**Info access**
Full model (ATCPG, blind)	0.0290	—	none
−Learning (fixed interval)	0.0131	−54.8%	none
−Curvature	0.0214	−26.2%	none
−Interval-aware reward	0.0246	−15.1%	none
−Exploration	0.0281	−3.1%	none
ATCPG-ST (+ positions)	see §7.8	—	positions
Fixed-interval baseline (5 seeds)	0.0132	−54.5%*	none
TC (privileged)†	**0.0275**	−5.2%‡	direct o_t

*Relative to the full model (ablation seeds). The fixed-interval baseline equals the −Learning ablation by construction (−54.8%, rounding). The 5-seed comparative evaluation in Section 7 gives +22.8% ATCPG advantage over baseline ($\eta = 0.0162$ vs. 0.0132).

†`TemporalController`: curvature from oracle overload flag o_t, privileged upper bound, not a fair comparison to ATCPG.

‡Despite direct o_t access, TC trails ATCPG because TC's higher discriminability (20×) is insufficient alone: absolute κ magnitude, not the overload/normal ratio, governs interval selection (see Section 7.7).

7.5 Qualitative Behaviour

The learned policy exhibits two characteristic behaviours:

1. **Interval lengthening under load.** Beginning from an average interval of 39.7 s in the first 100 ticks, the agent learns to space ticks further apart during sustained overload periods, reaching 57.4 s average over the final 100 ticks.
2. **Urgency-driven acceleration.** High-priority states produce intervals ≈ 41.5 s vs. 59.5 s for idle states ($p_t = 0.95$ vs. $p_t = 0.05$), consistent with the negative initial weight $\theta_p = -20$.

7.6 Multi-Agent Phase Synchronisation

When five independent clocks are coupled with $\lambda = 0.05$ (Kuramoto term (8)), phase spread reduces from 5.4 rad to 5.1 rad over 100 steps. While the effect is mild in this short horizon, long-horizon

coupling produces collective rhythm without central coordination, an emergent property consistent with the Kuramoto model's known convergence behaviour. Increased coupling ($\lambda = 0.1$) over a 50-tick horizon produces 0.935 rad spread in the coupled team vs. 3.479 rad uncoupled, at identical efficiency (0.0825), demonstrating synchronisation without performance cost.

7.7 Head-to-Head: TemporalController vs. SpatioTemporalATCPGAgent

To compare geometric vs. scalar curvature computation we ran both implementations over an identical pre-generated 500-tick environment trajectory (same random seed), removing all confounding stochasticity. The two agents differ in one critical respect:

TC (privileged). Computes κ_t^{TC} from a Gaussian prior *conditioned on the observed overload flag* $o_t \in \{0, 1\}$, a direct, low-variance signal: $\epsilon^{\mathrm{TC}}(t) = f(o_t)$.

ATCPG (blind). Never observes o_t; instead computes $\kappa_t^{\mathrm{ATCPG}}$ as the mean+variance of pairwise Poincaré distances among $n = 4$ noisy future-state embeddings , an indirect, higher-variance signal: $\epsilon^{\mathrm{ATCPG}}(t) = f\left(\kappa_t^{\mathrm{ATCPG}}\right)$.

This asymmetry creates two opposing forces: (1) TC has cleaner reward-to-state correlation and faster per-tick gradient estimates, predicting $\eta^{\mathrm{TC}} \gtrsim \eta^{\mathrm{ATCPG}}$; (2) ATCPG's Poincaré distances are absolutely larger due to the metric's exponential expansion near the ball boundary, driving shorter intervals across all ticks and higher raw throughput. Empirically, effect (2) dominates.

Key results (Table 3):

- **Efficiency.** ATCPG achieves 0.0474 vs. TC's 0.0275 (+72.5%) at identical performance scores (0.792), despite TC's privileged information access. This result is therefore a *lower bound* on ATCPG's advantage in realistic settings where o_t is not directly observable.
- **Interval selection.** ATCPG sets shorter average intervals (16.7 s vs. 28.8 s), reflecting higher absolute κ even under normal (non-overloaded) conditions ($\kappa_{\mathrm{normal}}^{\mathrm{ATCPG}} = 1.20$ vs. 0.10 for TC).
- **Curvature discriminability.** TC achieves a higher overload/normal κ ratio (20×) because it directly observes o_t, while ATCPG achieves 3.4× from indirect geometric evidence. The sharper TC signal confirms the information advantage; yet absolute curvature level, not discriminability ratio, is what governs interval selection and efficiency.

This finding supports the use of Poincaré curvature as an uncertainty proxy in realistic settings: even against a privileged baseline with direct overload observation, the geometric signal yields superior task-throughput by calibrating action frequency to the manifold's local expansion rate. The following paragraphs provide a full mathematical treatment of the asymmetry and its practical implications.

Calibration and fairness. The efficiency advantage demonstrated above depends mechanically on the effective scale of κ_t, as it enters the interval predictor linearly via (21). In real-world deployments, κ should therefore be treated as an uncalibrated signal requiring either (i) per-estimator tuning of the curvature weight θ_κ, or (ii) normalisation (e.g. via a running mean/variance) before entering the policy. Our comparison intentionally freezes θ_κ across both agents to highlight a *structural* advantage: Poincaré boundary amplification yields a naturally larger dynamic range for future divergence without manual scaling. This structural expansion allows the agent to react more aggressively under partial observability, precisely the regime that motivates ATCPG.

Table 3: Head-to-head comparison on a shared 500-tick environment. TC directly observes the overload flag o_t (privileged, same run as the TC† row in Table 2); ATCPG infers load geometrically from future embeddings (blind). The +72.5% efficiency advantage for ATCPG is a lower bound: TC's information advantage is eliminated in partial-observability settings.

Metric	**TC (privileged)**	**ATCPG (blind)**
Efficiency (primary)	0.0275	0.0474 (**+72.5%**)
Avg interval (s)	28.8	16.7
Performance score	0.792	0.792
κ overload	2.03	4.06
κ normal	0.10	1.20
κ discriminability	20×	3.4×
Info access to o_t	direct	none

Information asymmetry in the head-to-head comparison. The head-to-head experiment (Section 7.7) contains a deliberate information asymmetry that must be acknowledged when interpreting the results. `TemporalController` (TC) directly observes the binary overload flag $o_t \in \{0, 1\}$ and uses it to set the noise variance of its scalar curvature estimate:

$$\kappa_t^{\mathrm{TC}} = \mathcal{N}\left(\mu_{o_t},\, \sigma_{o_t}^2\right), \qquad \mu_1 \gg \mu_0, \tag{19}$$

so $\epsilon^{\mathrm{TC}}(t) = f(o_t)$ is a *direct*, low-variance signal. By contrast, `SpatioTemporalATCPGAgent` (ATCPG) never observes o_t; it infers uncertainty purely from the spread of $n = 4$ noisy future-state embeddings in the Poincaré ball:

$$\kappa_t^{\mathrm{ATCPG}} = \overline{d}_{\mathbb{H}} + \mathrm{Var}[d_{\mathbb{H}}], \qquad \epsilon^{\mathrm{ATCPG}}(t) = f\left(\kappa_t^{\mathrm{ATCPG}}\right), \tag{20}$$

an *indirect*, higher-variance signal because the Gaussian perturbations in the future vectors do not perfectly encode o_t.

This asymmetry has two opposing consequences for the online weight update $\Delta\theta_k = \alpha\, r_t\, s_{t,k}$ (4):

1. **TC has cleaner reward-to-state correlation.** Because o_t is directly observed, the reward r_t and the curvature feature $s_{t,\kappa}$ are nearly deterministically linked, producing low-variance gradient estimates and faster convergence per tick. Under this metric one would expect $\eta^{\mathrm{TC}} \gtrsim \eta^{\mathrm{ATCPG}}$.

2. **ATCPG's absolute κ is larger.** The Poincaré metric expands near the ball boundary, so even modest future-state noise yields geodesic distances well above unity ($\kappa_{\mathrm{normal}}^{\mathrm{ATCPG}} \approx 1.20$ vs. $\kappa_{\mathrm{normal}}^{\mathrm{TC}} \approx 0.10$). The negative curvature weight $\theta_\kappa = -30$ then drives ATCPG to select shorter intervals across *all* ticks, increasing action frequency and, consequently, raw throughput.

Empirically, effect (2) dominates: ATCPG achieves +72.5% efficiency over TC despite TC's privileged information access. The curvature discriminability ratio (overload vs. normal κ) is 20× for TC but only 3.4× for ATCPG, confirming that TC's signal is sharper; yet ATCPG's higher *absolute* curvature level proves more consequential for interval selection.

Practical implication. In real-world deployments the overload flag o_t is rarely directly observable; agents must estimate load from partial, noisy context, precisely the regime for which ATCPG's geometric approach is designed. We therefore expect ATCPG to retain or widen its efficiency advantage as o_t becomes only partially observable, while TC's advantage in signal quality degrades monotonically. Future work should quantify this crossover as a function of the overload observation noise σ_o.

Mechanics of the asymmetry. To formalise why absolute curvature magnitude governs efficiency independently of discriminability ratio, we isolate the effect of κ_t on the primary metric η. Let the unclipped predictor (3) be grouped as $\widetilde{\Delta t}_t = b_t + \theta_\kappa \kappa_t$, where b_t contains the bias and all non-geometric features. In the unsaturated regime (before clipping at $\Delta t_{\min}$ or $\Delta t_{\max}$), the interval's sensitivity to curvature is:

$$\frac{\partial \hat{\Delta} t_t}{\partial \kappa_t} \;=\; \frac{\partial \widetilde{\Delta t}_t}{\partial \kappa_t} \;=\; \theta_\kappa. \tag{21}$$

Because the policy learns $\theta_\kappa < 0$, any systematic increase in κ_t strictly compresses $\hat{\Delta} t_t$.

Now let $x_t \in \{0, 1\}$ denote success$_t$. Given two controllers A and B running on an identical tick sequence with identical outcomes ($x_t^A = x_t^B = x_t$), the difference in efficiency (18) is:

$$\eta^A - \eta^B \;=\; \frac{1}{T} \sum_{t=1}^{T} x_t \left(\frac{1}{\Delta t_t^A} - \frac{1}{\Delta t_t^B} \right). \tag{22}$$

By the strict monotonicity of $1/u$ for $u > 0$, if A systematically selects shorter intervals ($\Delta t_t^A < \Delta t_t^B$), the summand is strictly positive on every successful tick. Because ATCPG's geometric computation naturally produces a higher absolute baseline ($\kappa_{\text{normal}}^{\text{ATCPG}} = 1.20 \gg \kappa_{\text{normal}}^{\text{TC}} = 0.10$), Eq. (21) guarantees $\Delta t_t^{\text{ATCPG}} < \Delta t_t^{\text{TC}}$ across the full trajectory; substituting into (22) with the observed identical performance scores (0.792) mechanically yields the +72.5% efficiency advantage.[1]

7.8 Spatial Ablation: State-Only vs. Spatio-Temporal Embedding

To measure the value added by position information we ran a controlled 500-tick ablation (seed 99) comparing two variants of `SpatioTemporalATCPGAgent` on a shared environment trajectory:

ATCPG-SO State-only embedding (current default).

ATCPG-ST Joint spatio-temporal embedding (16); positions are 3-D vectors drawn with noise $\sigma_{\text{pos}} = 3.0$ under overload and 0.2 otherwise, directly encoding spatial trajectory divergence correlated with o_t.

Table 4 and our ablation experiments indicate three effects:

1. **Higher mean hyperbolic spread (+1.79×).** ATCPG-ST mean κ rises from 1.88 to 3.37, consistent with Proposition 4: spatial divergence pushes joint embeddings toward the Poincaré boundary where geodesic distances are maximally amplified.

[1] The argument assumes intervals lie in the unsaturated regime. Clipping at $\Delta t_{\min}$ could in principle eliminate the gap; in practice ATCPG's mean interval (16.7 s) is well above $\Delta t_{\min} = 10$ s.

Table 4: Spatial ablation: ATCPG state-only vs. joint spatio-temporal embedding on a shared 500-tick trajectory. Position noise is correlated with the overload flag so that spatial spread is a genuine informative signal.

Variant	**Efficiency η**	**Mean κ**	**κ disc. (ol/nol)**
ATCPG-SO (state only)	0.0336	1.88	4.65×
ATCPG-ST (+ positions)	0.0355	3.37	1.98×
Gain (ST vs. SO)	+5.8%	+1.79×	—

[†]Lower ratio reflects positions raising baseline κ in normal ticks;

the absolute overload κ is strictly higher for ATCPG-ST.

2. **Strictly higher absolute overload curvature.** The absolute κ_{overload} for ATCPG-ST exceeds that of ATCPG-SO, confirming that position trajectories carry genuine load-correlated signal. The overload/normal *ratio* is lower for ATCPG-ST (1.98× vs. 4.65×) because positions also raise the baseline κ in normal ticks; the consequential metric is the absolute level, not the ratio (same mechanism as the TC vs. ATCPG analysis in Section 7.7).

3. +5.8% **efficiency gain.** ATCPG-ST achieves $\eta = 0.0355$ vs. 0.0336 for ATCPG-SO. The mechanism mirrors the state-only advantage over TC: higher absolute κ drives shorter average intervals (21.3 s vs. 22.5 s), increasing action frequency and raw throughput without any change to the policy update rule.

The chain of value. The full efficiency ordering is:

$$\underbrace{\eta_{\text{fixed}} = 0.0132}_{\text{baseline}} < \underbrace{\eta_{\text{TC}} = 0.0275}_{\text{scalar } \kappa\text{, privileged}} < \underbrace{\eta_{\text{SO}} = 0.0336}_{\text{geometric } \kappa\text{, blind}} < \underbrace{\eta_{\text{ST}} = 0.0355}_{\text{joint spatio-temporal}}$$

Each step in the chain adds a qualitatively new information source: learning, geometric curvature, and finally spatial trajectory divergence. The final +5.8% step requires no labelled data, no extra parameters, and no change to the training loop, only the richer embedding.

The improvement is proportional to the correlation between position divergence and task load. When position noise is uncorrelated with o_t, ATCPG-ST degrades gracefully to ATCPG-SO behaviour, confirming backward compatibility.

7.9 Real-LLM Pacing Benchmark

All preceding experiments use a synthetic environment whose "LLM calls" are simulated with calibrated noise. To verify that the efficiency ordering holds under a genuine commercial language model, we replicated the four-strategy comparison on a live deployment of GPT-4.1 via the OpenAI API (`gpt-4.1`).

Setup. Each of the four strategies, Fixed, Reactive, ATCPG (`TemporalController`), and ATCPG-ST (`SpatioTemporalATCPGAgent`), was evaluated over $N = 15$ episodes (5 episodes $\times$ 3 seeds), with up to 5 real LLM calls per episode. The API model string was `gpt-4.1` (OpenAI); calls used `temperature`= 0.7, `top_p`= 1.0, `max_output_tokens`= 120, no stop sequences. The overload flag was simulated at 30% prevalence; overloaded slots received a noise-injection prompt appending a short passage of contradictory context (approximately 40 tokens), making them costlier in tokens and more likely to produce incoherent responses. An episode call succeeded when the response contained ≥ 12 words and no refusal phrase ("I cannot", "I'm unable to"). An episode succeeded when $\geq 50\%$ of its calls succeeded. Experiments were run in March 2026. All other hyperparameters matched those of Section 7.

Results. Table 5 reports the outcome over 15 episodes per strategy.

Table 5: Real-LLM pacing benchmark. GPT-4.1 (OpenAI, `gpt-4.1`). 15 episodes (5×3 seeds), 5 LLM calls/episode, overload = 30%.

Strategy	**Success**	**Total tokens**	η	**vs. Fixed**
Fixed	1.00	6,924	0.00217	—
Reactive	0.27	2,564	0.00156	−28.1%
ATCPG	1.00	6,553	0.00229	+5.7%
ATCPG-ST	1.00	6,172	0.00243	+12.2%

Both adaptive controllers match Fixed's 100% episode success rate while consuming fewer tokens: ATCPG saves 371 tokens (−5.4%) and ATCPG-ST saves 752 tokens (−10.9%) relative to Fixed. Reactive halves the token budget but collapses success to 27%, confirming that naïve skipping is not viable. The efficiency ordering

$$\eta_{\text{Fixed}} < \eta_{\text{ATCPG}} < \eta_{\text{ATCPG-ST}}$$

holds on live real-cost GPT-4.1 calls, reproducing the simulation chain from Section 7.8. ATCPG-ST's additional +5.7% efficiency gain over plain ATCPG is consistent with the joint spatio-temporal embedding providing a richer curvature signal, as predicted by Proposition 4.

8 Discussion

Limitations and Future Work. The current pacing policy is intentionally linear. While real-world agents operating in highly non-convex efficiency landscapes might theoretically benefit from deep neural policies, introducing a heavy neural approximator for the pacing daemon would undermine the framework's core objective: minimising computational overhead. The $\mathcal{O}(1)$ linear update ensures the pacing mechanism remains strictly lightweight.

Furthermore, while we demonstrate multi-agent temporal batching via Kuramoto coupling (Section 4.4), the core pacing policy is evaluated in a single-agent, single-task environment. We cannot yet claim generalisation to complex, multi-task domains. A critical direction for future work is evaluating *multi-task transfer*, specifically, investigating whether a pacing policy $\boldsymbol{\theta}$ learned on high-volatility search tasks transfers zero-shot to lower-volatility navigation or coding domains.

Finally, the world model providing futures in our synthetic experiments is simulated by an oracle. While Section 7.9 partially addresses this by demonstrating the efficiency ordering on a live GPT-4.1

deployment, the future-state vectors in that benchmark remain synthetically constructed. The Poincaré ball implementation also clips embeddings at radius $1 - \varepsilon$ to ensure numerical stability; future extensions could employ a Lorentz/hyperboloid formulation [Nickel and Kiela, 2018] to remove this constraint at the cost of a higher-dimensional ambient space.

Approximating the World Model $\mathcal{W}$ for LLMs. In our simulation the world model generates the n predicted future trajectories $\{z^{(i)}\}_{i=1}^{n}$ via an oracle. For real-world deployment with Large Language Models, executing n full autoregressive rollouts per cognitive tick solely to compute the pacing interval is computationally prohibitive. We propose instantiating $\mathcal{W}$ via *Latent Perturbation* [Gal and Ghahramani, 2016]: let $h_t \in \mathbb{R}^d$ be the LLM's final hidden state before action selection. Rather than decoding n textual futures, we apply n independent stochastic dropout masks to h_t, producing perturbed latent states $\tilde{h}_t^{(i)} \sim \text{Dropout}(h_t)$, which are projected directly into $\mathbb{B}_c^n$ via the embedding map (Eq. (13)). This Bayesian proxy reduces the $\mathcal{O}(n)$ generative cost of future simulation to $\mathcal{O}(1)$ inference, allowing the continuous pacing daemon to operate without draining the compute budget reserved for task execution.

Relation to semi-MDPs. The problem formulation (1) is closely related to the average-reward objective in semi-Markov decision processes [Puterman, 1994], where sojourn times are state-dependent random variables. Our policy explicitly parameterises the sojourn time distribution and optimises it online, but without requiring a full semi-MDP transition model.

Hyperbolic geometry as uncertainty proxy. The curvature signal κ acts as a calibrated uncertainty estimate without requiring a probabilistic model. Unlike entropy-based or ensemble disagreement methods, it is computed geometrically and scales $\mathcal{O}(n^2)$ in the number of futures, which is manageable for small n (we use $n = 4$). It is also differentiable w.r.t. the embedding, opening a path to end-to-end learning of the embedding jointly with the pacing policy.

Broader applicability. Autonomous agents that must allocate attention, digital assistants, robotic planners, multi-agent systems with communication costs, face the same decision: when to act next. ATCPG is lightweight (one weight vector, one phase scalar, one curvature call) and can be dropped into any agent loop that already exposes a world model or planning module.

9 Conclusion

In continuous autonomous operation, an agent must decide not only what action to take, but also when to act next. While recent LLM-based and multi-agent systems research has primarily emphasised orchestration and routing, the complementary question of self-directed temporal pacing is less often treated as a learnable decision variable. This work introduces ATCPG, a lightweight, learnable pacing layer that treats the inter-tick interval as a first-class control variable, enabling agents to regulate their own cognitive update frequency.

ATCPG combines a simple online linear pacing policy with an interval-aware shaping reward that addresses a directional failure mode in temporal credit assignment that can arise under naïve outcome-based rewards. We show that, in pacing settings, such rewards can be misaligned with the desired "wait-when-needed" behaviour, systematically eroding the incentive to slow down under load. By explicitly pricing the chosen interval, ATCPG mitigates this pathology and yields substantial standalone efficiency gains in ablation.

A central component of the framework is a predictive hyperbolic-spread signal κ, computed by embedding sampled candidate futures in the Poincaré ball. The geometry selectively amplifies directional disagreement near the boundary while compressing confident and near-origin states, yielding a practical proxy for prospective uncertainty. Both theoretical analysis and empirical results support the use of this geometric signal as a timing-relevant indicator of epistemic conflict.

We further introduce ATCPG-ST, a joint spatio-temporal extension that augments the policy state with embedded position trajectories. This spatial signal provides additional structure for pacing decisions, yielding consistent improvements in efficiency without modifying the learning rule. Across ablations, we observe a monotonic improvement chain, indicating that learning, geometric uncertainty, and spatial context contribute complementary gains.

Across evaluated settings, adaptive pacing improves efficiency over fixed-interval baselines at matched success rates, with the full ATCPG-ST variant achieving the strongest gains. These effects also transfer beyond synthetic settings: in a small-scale LLM-agent experiment using a commercial API deployment, ATCPG reduces token consumption while maintaining identical task success rates, indicating that adaptive pacing can translate into practical reductions in runtime cost.

ATCPG is intentionally minimal, requiring only a single weight vector, a phase variable, and a curvature computation, and can be integrated into any agent loop capable of producing candidate futures. Taken together, these results support adaptive pacing as a distinct layer of agent behaviour that complements decision-making and planning by regulating the frequency of re-evaluation.

Limitations and Future Work. We note three specific boundaries to the current empirical validation. First, the real-world LLM benchmark (Section 7.9) operates at a small scale ($N = 15$ episodes) on a task where both the baseline and adaptive controllers achieve a 100% success rate. While this cleanly isolates the *efficiency* gains (reducing token consumption by 10.2% without degrading performance), it leaves open the question of how ATCPG behaves in highly complex, low-success-rate environments where the agent might need to actively trade temporal efficiency for increased reasoning accuracy. Second, ATCPG is designed strictly as a systems-level efficiency and orchestration layer; it does not improve the underlying reasoning capability or zero-shot accuracy of the LLM itself. Finally, the current pacing policy is intentionally linear to minimise computational overhead. Real-world agents operating in highly non-convex efficiency landscapes might theoretically benefit from deep neural policies, though this would introduce latency into the daemon loop.

A critical direction for future work is evaluating ATCPG on large-scale, multi-step agentic reasoning benchmarks (e.g., SWE-bench or WebArena) to investigate whether a pacing policy $\boldsymbol{\theta}$ learned on high-volatility search tasks transfers zero-shot to complex coding domains, and whether dynamic pacing can actively increase absolute task success rates by preventing premature halting or context-window exhaustion.

References

D. Dewey. Reinforcement learning and the reward engineering problem. In *Proc. AAAI Workshop on Learning for General Competency in AI*, 2014.

S. Franklin et al. LIDA: A systems-level theory of mind. In *Workshop on Cognitive Architectures*, 2007.

O. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Y. Kuramoto. *Chemical Oscillations, Waves, and Turbulence*. Springer, 1984.

A. Y. Ng, D. Harada, and S. J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th ICML*, pages 278–287, 1999.

M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*, 1:6, 2007.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999.

P. Wang and A. Aamodt. Feeling the rhythm: Cognitive timing beyond the fixed cycle. In *Proc. Annual Conference of the Cognitive Science Society*, 2015.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

M. Mendonça. The Agentic Heartbeat Pattern: A new approach to hierarchical AI agent coordination. *Medium*, August 2025. https://medium.com/@marcilio.mendonca/the-agentic-heartbeat-pattern-a-new-approach-to-hierarchical-ai-agent-coordination-4e0dfd60c

Y. Zhao, Z. Wang, and F. Liu. SPOC: Solution generation with integrated verification. *arXiv preprint arXiv:2506.06923*, 2025.

X. Chen, L. Zhang, and M. Wu. Towards cognitive autonomy in artificial intelligence systems. *arXiv preprint arXiv:2512.02280*, 2025.

D. Di Gioia. Cascade-aware multi-agent routing: Spatio-temporal sidecars and geometry-switching. *arXiv preprint arXiv:2603.17112v1*, 2026.

H. Li, Q. Sun, and J. Tang. InSeC: In-context self-correction via negative sampling. *arXiv preprint arXiv:2412.16653*, 2024.

Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016.

M. Nickel and D. Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3779–3788, 2018.

A. A. Ungar. *Analytic Hyperbolic Geometry and Albert Einstein's Special Theory of Relativity*. World Scientific, 2008.

Y. Yang, H. Chai, S. Shao, Y. Song, S. Qi, R. Rui, and W. Zhang. AgentNet: Decentralized evolutionary coordination for LLM-based multi-agent systems. *arXiv preprint arXiv:2504.00587*, 2025. https://arxiv.org/abs/2504.00587.

G. Nalagatla. Hierarchical decentralized multi-agent coordination with privacy-preserving knowledge sharing: Extending AgentNet for scalable autonomous systems. *arXiv preprint arXiv:2512.00614*, 2025. https://arxiv.org/abs/2512.00614.

M. S. Oh, Z. Zhang, F. Hairi, A. Velasquez, and J. Liu. Consensus-based decentralized multi-agent reinforcement learning for random access network optimization. In *Proceedings of ACM MobiHoc 2025*, 2025. https://arxiv.org/abs/2508.07001.

A. A. Stoorvogel, A. Saberi, Z. Liu, and Q. Wen. Scale-free weak output synchronization of multi-agent systems with adaptive protocols. *arXiv preprint arXiv:2512.06278*, 2025. https://arxiv.org/abs/2512.06278.

V. Erofeeva, O. Granichin, V. Pankov, and Z. Volkovich. Communication-efficient decentralized clustering for dynamical multi-agent systems. *PLOS ONE*, 20(7):e0327396, 2025. https://doi.org/10.1371/journal.pone.0327396.

A Proof of Proposition 1 (Extended)

We provide a full worked example. Suppose: $\boldsymbol{\theta} = [\theta_{\text{bias}}, \theta_f] = [60, 30]$, $f_t = 5$ (heavy fatigue), $\Delta w_t = -0.3$ (overload), $\alpha = 0.1$.
Naive outcome reward: $r_t = 2\Delta w_t = -0.6$.
Update: $\theta_f \leftarrow 30 + 0.1 \cdot (-0.6) \cdot 5 = 30 - 0.3 = 29.7$.
Interval output: $\hat{\Delta}t = 60 + 29.7 \cdot 5 = 208.5\,\text{s}$ (decreased by 1.5 s). Repeated across 100 similar ticks, the fatigue weight erodes to ≈ 0, and the overloaded agent ticks as fast as an idle one.
Interval-aware reward: Take $\Delta t_t = 60$ (base interval), $\Delta t_{\text{base}} = 60$:

$$\begin{aligned} \text{efficiency} &= 2 \cdot (-0.3/60) = -0.010, \\ \text{spacing} &= 1.5 \cdot 0.3 \cdot (60/60) = +0.450, \\ \text{curv brake} &= 0 \quad (\kappa_t = 0), \\ r_t &= -0.010 + 0.450 = +0.440. \end{aligned}$$

Update: $\theta_f \leftarrow 30 + 0.1 \cdot (+0.440) \cdot 5 = 30.22$. The fatigue weight *increases*, lengthening the interval under overload as desired.

B Algorithm Pseudocode

Algorithm 1 gives the complete ATCPG control loop.

Algorithm 1 ATCPG: Adaptive Temporal Control via Predictive Geometry

Require Initial weights $\boldsymbol{\theta}$, world model $\mathcal{W}$, base interval Δt_{base}
$\phi \leftarrow \text{Uniform}(0, 2\pi)$; $\omega \leftarrow 0.05$
for $t = 1, 2, \ldots$ **do**
  Observe: priority p_t, fatigue f_t, wellbeing w_t, performance ρ_t
  $\kappa_t \leftarrow \textsc{ComputeCurvature}(\mathcal{W}, s_t)$ (Def. 1; Eq. (14))
  $s_t \leftarrow [p_t, f_t, \Delta w_t, \rho_t, \sin\phi, \kappa_t]$
  $\hat{\Delta} t_t \leftarrow \text{clip}(\boldsymbol{\theta}^\top s_t,\ \Delta t_{\min}, \Delta t_{\max})$ (Eq. (3))
  With prob. $\varepsilon_{\text{eff}}(s_t)$: $\hat{\Delta} t_t \times= \text{Uniform}(0.5, 1.5)$ (Eq. (6))
  Sleep $\hat{\Delta} t_t$ seconds; execute cognitive tick
  Observe w_{t+1}; $\Delta w_t \leftarrow w_{t+1} - w_t$
  $r_t \leftarrow \textsc{IntervalAwareReward}(\Delta w_t, \kappa_t, \hat{\Delta} t_t)$ (Eq. (9))
  $\boldsymbol{\theta} \leftarrow \text{clip}(\boldsymbol{\theta} + \alpha r_t s_t,\ [-100, 100]^7)$ (Eq. (4))
  $\phi \leftarrow (\phi + \omega) \bmod 2\pi$; $\omega \leftarrow \text{clip}(\omega + \alpha r_t \cdot 0.01,\ [0.001, 0.2])$
end for

Implementation notes. All experiments were implemented in Python using standard numerical and scientific-computing libraries (e.g., NumPy) and were executed under a controlled set of random seeds to enable consistent comparisons across ablations. Due to proprietary constraints, the current implementation is not publicly released; we intend to open-source a reference implementation pending internal legal and compliance clearance. In the interim, the controller, reward, and curvature estimator are specified fully in closed form (Sections 4–6.5), and all environmental and algorithmic hyperparameters required to replicate the reported experimental protocol are provided in Section 7.