φ-Scheduler: Golden-Ratio Phase Scheduler for Multi-Actuator Control with Periodic Invariance, Interference Bounds, and Compliance API

Provisional Patent Draft

Field

This disclosure relates to real-time scheduling for multi-actuator control systems. It provides a scheduler that enforces *φ-commensurate* update windows, guarantees *periodic invariance* and *interference bounds* between actuators, and exposes a *compliance API* that any controller (including MPC or RL) can use while proving adherence to the schedule. The scheduler is domain-agnostic and applies to fusion devices, high-power lasers, robotics, beamlines, power electronics, and other multi-actuator plants.

Background

Multi-actuator systems suffer cross-interference when control updates align unfavorably with plant modes. Conventional schedulers use fixed or equal-spaced slots that can reinforce low-order modal coupling and lack portable guarantees or machine-verifiable compliance. There is a need for a reusable module that (i) desynchronizes updates via irrational-ratio timing, (ii) guarantees periodic invariance for control proofs and certification, (iii) bounds interference, and (iv) provides a simple API and audit trail.

Summary

A *φ-scheduler* partitions each control period into phase windows with durations related by the golden ratio $\varphi = \frac{1+\sqrt{5}}{2}$, assigns actuators to phase sets, and *enforces* that setpoint updates occur only within assigned windows. The schedule is *periodic invariant* with declared period and jitter bounds. For band-limited plant/actuator couplings, the scheduler yields a *quantified interference bound* on time-averaged bilinear cross-terms. A *compliance API* and cryptographically signed logs allow third-party controllers to plug in while proving conformity.

Definitions

φ-commensurate windows. Let $T > 0$ be a control period. Define windows $W_0, \ldots, W_{L-1}$ with durations $\Delta t_\ell > 0$ such that

$$\sum_{\ell=0}^{L-1} \Delta t_\ell = T, \qquad \frac{\Delta t_{\ell+1}}{\Delta t_\ell} \in \{\varphi, \varphi^{-1}\} \text{ for admissible } \ell,$$

with wrap-around interpreted modulo L and optional subperiod patterns (e.g. $L = 8$).

Actuator phase sets. Each actuator a is assigned $\Pi(a) \subseteq \{0, \ldots, L-1\}$. Updates to a may be applied only during windows W_ℓ with $\ell \in \Pi(a)$ and must satisfy dwell and slew constraints.

Periodic invariance. The schedule repeats every T with bounded jitter ϵ_j and supports periodic terminal sets/arguments in control proofs.

Interference bound (qualitative). For band-limited cross-coupling kernels between actuators, the scheduler ensures a strict bound

$$\langle \text{cross-term} \rangle \leq \kappa \, \langle \text{baseline cross-term} \rangle,$$

with $0 < \kappa < 1$ determined by window smoothness and spectral separation, providing a guaranteed reduction relative to co-phased or equal-spaced updates.

Compliance API. A programmatic interface that (i) issues window entry/exit events, (ii) accepts or rejects update requests according to $\Pi(a)$ and timing, and (iii) produces signed compliance records.

Detailed Description of Embodiments

Scheduler Core

Given T and L, the scheduler constructs $\Delta t_{0:L-1}$ satisfying φ-commensurability and maps actuators a to $\Pi(a)$. A high-resolution clock triggers *BeginWindow(ℓ)* and *EndWindow(ℓ)*. Update requests *RequestUpdate(a, payload)* are admitted only if the current window index $\ell \in \Pi(a)$ and actuator dwell/slew constraints are satisfied.

Periodic Invariance and Multi-Rate

The scheduler supports superframes of duration ST with $S \in \mathbb{N}$, preserving φ-relations within each T. Jitter ϵ_j is bounded by hardware timers or RTOS guarantees, enabling periodic MPC/RL safety proofs with phase-dependent terminal sets.

Interference Bounds

For plants with dominant modal content below a cutoff, the scheduler's irrational timing and window smoothness avoid low-order reinforcement. Over each period (or superframe), the time-averaged magnitude of bilinear cross-terms is reduced by a strict factor $\kappa \in (0, 1)$ relative to co-phased or equal-spaced baselines. The factor is scheduler-intrinsic and controller-agnostic.

Compliance and Audit

The compliance API exposes: *WindowIndex()*, *Allowed(a)*, *RegisterUpdate(a)*, and *GetComplianceReport()*. The report includes window timings, admitted/rejected updates, actuator

IDs, and cryptographic signatures, enabling certification and infringement testing. A hardware embodiment may use FPGA timers and secure elements for signing.

Example Domains

Fusion: gate NBI/ECRH/ICRH/RMP/pellets. *ICF:* gate sub-pulse timings. *Robotics:* desynchronize multi-axis torque updates. *Power electronics:* gate converter switching setpoint updates. *Beamlines/manufacturing:* stagger multi-actuator adjustments to reduce coupling.

Advantages

The module is plug-and-play, controller-agnostic, and portable. It guarantees schedule properties needed for control proofs, reduces cross-interference by design, and produces verifiable compliance artifacts.

Claims

Independent Claims

1. **(Method)** A method of scheduling updates in a multi-actuator control system, comprising:

   (a) partitioning each control period T into L phase windows $W_0, \ldots, W_{L-1}$ with durations Δt_ℓ satisfying $\sum_\ell \Delta t_\ell = T$ and $\Delta t_{\ell+1}/\Delta t_\ell \in \{\varphi, \varphi^{-1}\}$ for $\varphi = (1+\sqrt{5})/2$;

   (b) assigning each actuator a to a phase set $\Pi(a) \subseteq \{0, \ldots, L-1\}$ and enforcing that setpoint updates to a occur only during windows W_ℓ with $\ell \in \Pi(a)$;

   (c) guaranteeing periodic invariance by repeating the phase windows every T with bounded jitter; and

   (d) providing a compliance application programming interface (API) that exposes window state, admits or rejects update requests according to the schedule, and generates machine-verifiable compliance records.

2. **(System)** A scheduling system for multi-actuator control, comprising:

   (a) a timing engine that generates L φ-commensurate phase windows within each period T;

   (b) a gating module that maps actuators to phase sets and enforces update admissibility only within assigned windows and subject to dwell and slew constraints;

   (c) a periodicity manager that maintains repetition of the schedule with bounded jitter; and

(d) a compliance module providing an API for external controllers and producing signed compliance logs.

3. **(Non-transitory medium)** A non-transitory computer-readable medium storing instructions that, when executed by one or more processors, cause a system to perform the method of claim 1.

4. **(Method feature: interference bound)** The method of claim 1, further comprising ensuring that, for band-limited actuator cross-coupling, a time-averaged bilinear cross-term between actuators is bounded by a strict factor $\kappa \in (0, 1)$ relative to a baseline schedule lacking φ-commensurate windows.

Dependent Claims (Method of Claim 1)

5. The method of claim 1, wherein $L = 8$ and the window durations follow a repeating φ/φ^{-1} pattern.

6. The method of claim 1, wherein each actuator update further satisfies minimum dwell time, minimum inter-update separation, and maximum slew rate constraints enforced within windows.

7. The method of claim 1, wherein the schedule is rescaled by a reference timescale τ_{ref} while preserving φ-commensurability.

8. The method of claim 1, wherein a superframe of duration ST is used, preserving the window structure within each period T.

9. The method of claim 1, wherein the compliance API comprises at least: *BeginWindow(ℓ)*, *EndWindow(ℓ)*, *WindowIndex()*, *Allowed(a)*, *RequestUpdate(a, payload)*, and *GetComplianceReport()*.

10. The method of claim 1, wherein compliance records are cryptographically signed using a hardware security element or secure enclave.

11. The method of claim 1, wherein the interference bound is obtained by selecting window smoothness profiles that limit spectral leakage of the update sequence.

12. The method of claim 1, wherein the scheduler rejects or defers update requests that would violate phase assignments or jitter bounds and records such events in the compliance log.

Dependent Claims (System of Claim 2)

13. The system of claim 2, wherein the timing engine is implemented on an FPGA or real-time microcontroller with hardware timers.

14. The system of claim 2, wherein the compliance module exposes a controller-agnostic interface usable by model predictive controllers, reinforcement learning policies, or heuristic controllers.

15. The system of claim 2, wherein the system provides a verification service that reconstructs schedule adherence from time-stamped actuator update events.

Dependent Claims (Medium of Claim 3)

16. The non-transitory computer-readable medium of claim 3, wherein the instructions enforce actuator phase sets and jitter bounds even when external controllers issue conflicting update requests.

Exemplary Embodiments

Fusion: the scheduler gates neutral beam, RF heating, magnetic perturbation, fueling, and shaping updates to φ-phased windows. *ICF:* the scheduler emits φ-spaced sub-pulse triggers. *Robotics:* motor torque references are phase-staggered to reduce structural mode excitation. *Power electronics:* setpoint updates for converters are staggered to avoid beat-frequency amplification. In all cases, the compliance API logs conformance.

Abstract

A reusable φ-scheduler module partitions control periods into golden-ratio-commensurate phase windows, assigns actuators to phase sets, and enforces that updates occur only within assigned windows. The schedule is periodically invariant with bounded jitter and provides a qualitative interference bound that reduces cross-terms between actuators relative to baseline schedules. A compliance API admits or rejects update requests and produces machine-verifiable logs, enabling controller-agnostic integration across domains.