

# **Two-Regime Control: Latent Coordination vs Compensation in Intelligent Systems**

*Robotics / Embodied AI*

Tyson Jeffreys  
Independent Researcher  
tyson@staygolden.dev

Version 1.2 — February 11, 2026

**Series note (1/3).** This paper is Part 1 of a three-paper series on regime-level regulation in intelligent systems. Part 1 introduces a two-regime framing for embodied control (latent coordination vs. compensation) and proposes a measurable compensation index plus a slow baseline regulator layer. Part 2 extends the same regulation idea to representations via concept containers. Part 3 applies it to research systems via time-to-analysis and analysis-layer artifacts.

## Abstract

Intelligent embodied systems often operate in two qualitatively different control regimes. In **latent coordination**, distributed dynamics (mechanics, morphology, low-level reflexes, and low-bandwidth feedback) do most of the work, yielding low energy use and robust behavior. In **compensation**, behavior is maintained through costly overrides: increased stiffness, gain escalation, frequent replanning, estimator churn, and high-frequency corrective action. We argue that many real-world failures in robotics and embodied AI arise not primarily from poor local controllers, but from the absence of an explicit mechanism to **detect and regulate prolonged residence in compensation**.

We formalize this framing using a measurable **compensation index** derived from actuator effort, control-rate, estimation uncertainty, contact instability, and override activity. We then propose a slow **baseline regulator layer** ( $\approx 1\text{--}5$  Hz) that biases the system back toward latent coordination without micromanaging task control. The resulting architecture predicts reduced energy per task, lower peak actuator stress, faster recovery from perturbation, and improved robustness under distribution shift. We provide concrete implementation patterns, falsifiable experiments, and negative tests.

## 1. Introduction

Modern robotics and embodied AI systems routinely achieve impressive performance in constrained environments, yet exhibit a recurring pattern outside their comfort zone: **excessive corrective action** that is energetically costly and brittle. This manifests as high torque/thermal stress, jerky trajectories, contact slip, frequent replanning, instability under perturbations, and degradation under battery/compute constraints.

A common response is to improve the local controller, estimator, or planner. These improvements help—but they often miss a deeper architectural issue: the system lacks an explicit representation of **how control is being achieved** at a given moment.

This paper proposes a regime-level lens:

- **Latent coordination:** the system is well-coupled; low-cost dynamics carry the task.
- **Compensation:** the system is maintaining function through overrides and corrective contention.

Compensation is not “bad.” It is essential when resources drop or perturbations rise. The failure mode is **getting stuck** in compensation—quietly spending energy to hold together a system that is no longer coordinated.

Latent coordination is therefore a **supported regime**, not a willful state. When mechanical, estimator, or sensorimotor-coupling supports degrade, the system can preserve task output only by shifting into compensation. We refer to this transition pressure as a **support deficit**: insufficient structural support for low-cost coherence.

## 2. Two Control Regimes

### 2.1 Latent coordination

In latent coordination, behavior emerges from distributed alignment between:

- passive dynamics and morphology (compliance, inertial shaping)
- stable low-level primitives (oscillators, impedance/reflex loops)
- predictive coupling between sensing and actuation
- low-bandwidth feedback and minimal correction

Operational signatures:

- low corrective torque and low control-rate
- smooth trajectories; low jerk
- stable contact with low slip
- low estimator churn; uncertainty remains bounded
- low override frequency from higher-level arbitration

Intuitively: the system is in an **efficient basin** where small errors self-correct with little “fight.”

### 2.2 Compensation

In compensation, the system maintains function via corrective overrides:

- stiffness/gain escalation to suppress instability
- higher-frequency corrections (controller “chatter”)
- replanning bursts and safety arbitration
- aggressive sensing/estimation updates
- contact stabilization by force rather than alignment

Operational signatures:

- elevated actuator effort and thermal stress
- higher jerk or control derivative magnitude
- increased estimator covariance / prediction error spikes
- higher replanning/override duty cycle
- constraint-violation corrections (slip, saturation, resets)

Intuitively: the system is “holding together” by spending energy on correction rather than benefiting from coordinated dynamics.

## 3. Formal framing

Let the embodied system have physical state  $s(t)$  (e.g., joint positions/velocities, base pose, contact state) and control  $u(t)$  (e.g., torques, target accelerations, motor commands). Let  $\pi$  denote the task policy (learned or engineered) and  $\mathcal{C}$  denote constraints (stability, safety, contacts, limits).

We separate control into a low-cost coordination component and a higher-cost override component:

$$u(t) = u_0(s(t), \pi) + \Delta u(t)$$

- $u_0$  is the nominal low-level controller / primitive stack.
- $\Delta u$  represents corrective overrides from arbitration, replanning, gain escalation, and emergency stabilization.

We hypothesize that regime membership is best characterized not by task success alone, but by **internal regulatory cost**: how much corrective contention is required to maintain behavior.

#### 4. A measurable compensation index

We define a scalar **compensation index**  $C(t)$  as a weighted combination of measurable proxies:

$$C(t) = w_\tau \|\tau(t)\|^2 + w_{\dot{u}} \|\dot{u}(t)\|^2 + w_\Sigma \text{tr}(\Sigma(t)) + w_{\text{plan}} r_{\text{plan}}(t) + w_{\text{slip}} r_{\text{slip}}(t) + w_{\text{sat}} r_{\text{sat}}(t) + w_J u_J(t)$$

We treat learned **critics** / **judges** as *telemetry sources*, not authorities. In many deployments there is no ground-truth verifier for “is this action actually safe/efficient?” in the moment. A useful posture is therefore: let critics contribute uncertainty signals, and let **abstention** / **tie mass** tighten posture rather than forcing a brittle choice.

where, depending on platform:

- $\tau(t)$ : joint torque vector (or motor current proxy)
- $\dot{u}(t)$ : control derivative (or jerk proxy via commanded acceleration derivatives)
- $\Sigma(t)$ : estimator covariance (or prediction error proxy)
- $r_{\text{plan}}(t)$ : replanning / override rate (events per second)
- $r_{\text{slip}}(t)$ : slip rate / contact instability metric
- $r_{\text{sat}}(t)$ : actuator saturation / limit hits
- $u_J(t)$ : abstention / tie mass from learned critics (e.g., probability that one or more monitors cannot confidently rank an action or classify a contact state); treat high  $u_J$  as epistemic uncertainty that should tighten posture.

##### 4.1 Normalization

Raw units differ across terms. In practice, normalize each term by a baseline scale measured under stable operation:

$$\tilde{z}(t) = \frac{z(t)}{\mathbb{E}[z \mid \text{stable}] + \epsilon}$$

and compute  $C(t)$  using  $\tilde{z}$  to make weights interpretable.

##### 4.2 Persistence matters: filtered index and duty cycle

Compensation is not an instantaneous value; the pathological mode is sustained residence.

Define a low-pass filtered index  $\bar{C}(t)$ :

$$\dot{\bar{C}}(t) = \frac{1}{\tau_c} (C(t) - \bar{C}(t))$$

with  $\tau_c$  chosen to match “regime timescale” (e.g., 0.5–3 s).

Define a binary compensation indicator with hysteresis:

$$H(t) = \begin{cases} 1 & \bar{C}(t) > \theta_{\uparrow} \\ 0 & \bar{C}(t) < \theta_{\downarrow} \end{cases}$$

and the compensation duty cycle over a window  $[t_0, t_1]$ :

$$\text{DC}(t_0, t_1) = \frac{1}{t_1 - t_0} \int_{t_0}^{t_1} H(t) dt$$

Interpretation: -  $H(t) = 1$  indicates the system is in compensation. - DC measures how long it stays there.

## 5. The baseline regulator layer

We propose a slow regulator layer that shapes the system’s operating posture without micromanaging task control.

### 5.1 Regulator objective

Let  $L_{\text{task}}(s, u)$  represent task loss or tracking error. The regulator introduces an internal cost that penalizes sustained compensation:

$$J = \int_0^T \left( L_{\text{task}}(s(t), u(t)) + \lambda \phi(\bar{C}(t)) + \gamma \|\dot{\bar{C}}(t)\|^2 \right) dt$$

- $\phi$  is convex above a threshold (so high compensation is increasingly expensive).
- $\|\dot{\bar{C}}\|^2$  discourages thrash/oscillation.

### 5.2 What the regulator controls

The regulator acts on **meta-parameters**  $p(t)$  that modulate how the stack behaves:

- impedance / stiffness / damping gains
- planner update rate and horizon
- sensor polling rate and filter aggressiveness
- safety margins and risk posture
- “reset / recenter” routines (short stabilization protocols)

Critically: the regulator does not replace  $u_0$  or the planner; it biases them.

### 5.3 A simple control law

A practical regulator can be rule-based, learned, or hybrid. A minimal deterministic version:

1. Maintain a target band  $[C^-, C^+]$  for  $\bar{C}$ .
2. When  $\bar{C}$  rises persistently, invoke an **anti-compensation action** appropriate to the domain:
  - reduce unnecessary stiffness (when safe)
  - reduce planner micromanagement rate
  - execute a brief recenter routine (re-estimate, re-sync gait phase, settle posture)
  - adjust gait/stance parameters toward stable manifold

3. When  $\bar{C}$  returns to band, gradually relax posture back toward efficient settings.

Pseudocode sketch:

```

if barC > C_plus for longer than T_high:
    p <- p + alpha * delta_up(barC)      # increase safety posture / widen margins
    p <- p + beta * delta_recenter()      # short reset routine
elif barC < C_minus for longer than T_low:
    p <- p - eta * delta_down()          # relax toward efficient posture

apply_params(p)

```

The exact  $\delta$  operators are platform-specific; what matters is the architecture: slow regulation of posture using a measured internal-cost signal.

## 5.4 Why 1–5 Hz

Many compensation phenomena are not instantaneous. They are regimes that persist over hundreds of milliseconds to seconds:

- stiffness escalation persists after a perturbation
- planner override frequency stays elevated
- estimator covariance can remain high

A slow regulator prevents the system from “fighting itself” by reacting on the same timescale as the low-level controller.

## 5.5 Critics-as-sensors and abstention gating

In many real deployments there is no online “verifier” that can certify whether a novel contact, terrain interaction, or recovery maneuver is truly safe and efficient. When we add learned critics (from demonstrations, simulation, or logged rollouts), the correct stance is not to treat them as authorities, but as additional sensors.

Operationally, use **abstention / tie mass** as a first-class uncertainty signal. Let  $\bar{u}_J(t)$  be a low-pass filtered version of  $u_J(t)$  on the same regime timescale as  $\bar{C}(t)$ . When  $\bar{u}_J(t)$  rises, the regulator should *tighten posture* (reduce speed/force envelopes, widen safety margins, shorten horizons) and preferentially invoke low-impact routines (recenter, re-estimate, re-localize) until confidence returns. Persistent abstention should downgrade the system into a conservative controller or safe mode rather than forcing additional “clever” action.

A minimal gating rule (illustrative):

```

if  $\bar{C}(t)$  is high or  $\bar{u}_J(t)$  is high: tighten posture; prefer recenter/sense actions; add
hysteresis.
if either persists past a time limit: switch to conservative safety controller / stop.

```

Governance clause: learned critics can drift and can be gamed. They should be **versioned, monitored, and replay-tested** (a small fixed scenario suite) in the same way the telemetry→posture mapping is monitored. When critic health is unknown, degrade to conservative heuristics and envelopes (fail closed). **Phase-discipline reproducibility hook.** Treat irreversible writes/actuation as transition-window events: commit only in low-compensation windows, cap override duration,

and require rollbackable recovery when windows are missed. This “write when stable” discipline reduces irreproducible flailing and preserves auditability under perturbation. **Phase-discipline commitment-integrity hook.** If evidence is unchanged, posture must not silently revert. Any stance reversal in a commit window requires explicit change-basis logging (new evidence, discovered constraint, or explicit prior error), otherwise the regulator should withhold commit rights and route to gather/recenter. **Phase-discipline commit windows.** Commit windows require telemetry and should open only when strain and uncertainty signals are within bounded posture limits. The selection gate is the upstream posture controller: when telemetry degrades, it withholds commit rights and routes the system to evidence-gathering or recovery instead.

## 6. Why this improves energy and robustness

### 6.1 Convex energy argument

For many systems, energy/power increases nonlinearly with control effort (e.g., torque, current, switching losses). If power  $P$  is convex in effective effort  $e(t)$ , then reducing time spent at high effort yields disproportionate energy savings.

Let  $e(t)$  be an effort proxy correlated with  $\bar{C}(t)$ . Total energy:

$$E = \int_0^T P(e(t)) dt$$

If  $P$  is convex, then flattening spikes and reducing duty cycle in the high-effort regime reduces  $E$  more than proportional to mean effort reduction.

### 6.2 Robustness via internal cycling

Many brittle failures arise from chronic over-activation:

- saturation cascades
- estimator divergence under sustained churn
- contact instability amplified by high-frequency correction

A system that can downshift out of compensation faster will:

- recover quicker after disturbances
- reduce thermal/actuator stress accumulation
- avoid thrash loops that amplify errors

## 7. Experimental directions

The claims here are intended to be falsifiable with modest instrumentation.

### 7.1 Core prediction

At comparable task success, adding a baseline regulator that reduces compensation duty cycle should produce:

- 1) lower energy per task (or per meter)
- 2) lower peak torque/current and thermal stress proxies
- 3) fewer catastrophic failures under perturbation
- 4) faster recovery time after disturbances
- 5) smoother degradation under resource constraints (battery, compute)

## 7.2 Experiment A: locomotion under perturbation

**Setup:** biped or quadruped on variable terrain with external pushes and friction variation.

**Conditions:** - A1: baseline controller/planner only - A2: same controller + compensation index + baseline regulator

**Metrics:** - Joules/meter, peak current, temperature rise - slip/fall rate under perturbation - recovery time to stable gait - compensation duty cycle DC

**Negative test:** if DC decreases but energy does not, the claimed link between regime regulation and efficiency is weakened.

## 7.3 Experiment B: manipulation with contact uncertainty

**Setup:** insertion, door opening, or tool use with variable friction and compliance.

**Metrics:** - contact slip/impulse rate - jerk and force spikes - replanning/override bursts - success rate vs energy

**Negative test:** if regulator reduces compensation but increases failures or induces oscillatory cycling, the design must incorporate better hysteresis or context gating.

## 7.4 Experiment C: sensor degradation / latency shift

**Setup:** mobile robot or drone with induced sensor noise/latency.

**Hypothesis:** compensation manifests as estimator churn + control chatter.

**Metrics:** - covariance/prediction-error integrated - control-rate increase - energy and stability

## 8. Relation to existing control patterns

This paper does not claim novelty in hierarchical control or impedance control per se. The proposed contribution is the explicit **regime-level variable** and **slow regulatory layer**:

Viewed through a more general regulation lens,  $\bar{C}(t)$  (optionally augmented with  $\bar{u}_J(t)$ ) is simply a scalar *restraint* / *strain telemetry* that drives posture. The meta-parameters  $p(t)$  correspond to an operating band (tight vs loose envelopes), and the recenter/safe-mode routines correspond to explicit recovery transitions. In other words, this is an embodied instantiation of “regulation as a ground condition”: measure internal strain, gate risky actions under uncertainty, and bias the system back toward low-compensation basins.

- Hierarchical stacks exist, but often lack an explicit “time in compensation” signal.
- Safety layers exist, but often do not regulate *posture* back toward coordination once safe.
- Many systems can enter compensation; fewer can reliably and quickly exit.

## 9. Limitations and scope

- **Domain specificity:** the best  $C(t)$  terms differ by platform; a single universal index is unlikely.
- **Safety tradeoffs:** reducing stiffness is not always appropriate; the regulator must be constraint-aware.
- **Tuning:** poor hysteresis can introduce oscillatory cycling (a new kind of thrash).
- **Observability:** some systems lack sensors for reliable covariance/slip/torque estimation.
- **Critic governance:** learned critics (if used) can drift, be overconfident, or be adversarially induced. Treat them as telemetry; version and monitor them; maintain a replay-suite; and fail closed by tightening posture when critic health or agreement is poor.



## 10. Conclusion

We introduced a two-regime framing for embodied control: latent coordination versus compensation. Many failures and inefficiencies arise not because robots cannot compensate, but because they lack an explicit mechanism to detect and regulate prolonged compensation.

By defining a measurable compensation index and adding a slow baseline regulator layer, we obtain a falsifiable architectural hypothesis: **systems that reduce compensation duty cycle should use less energy and be more robust under perturbation at comparable task performance.**

**Cross-cutting principle.** Coherence requires support: regulation specifies and maintains the supports that keep behavior aligned, perturbation-tolerant, and recoverable.

---

## Appendix A: Practical implementation notes

### A.1 Computing $C(t)$ from logs

Minimal viable index for many robots:

$$C(t) = w_\tau \|\tau(t)\|^2 + w_{\dot{u}} \|\dot{u}(t)\|^2 + w_{\text{sat}} r_{\text{sat}}(t)$$

Add terms as available: - covariance/prediction error - slip/contact instability - replanning/override counters - critic abstention/tie mass  $u_J(t)$  (if using learned monitors)

### A.2 A simple “recenter” routine (platform-agnostic)

A recenter routine is a brief protocol to restore coupling:

- pause high-level corrections for a short window
- re-estimate state / reset filter if needed
- re-sync phase variables (gait/CPG) if applicable
- ramp impedance back smoothly

### A.3 Mapping to a general internal-load variable

If you already track an internal load  $x(t)$  (activation/effort), treat  $\bar{C}(t)$  as a robotics-specific estimator of  $x(t)$ .

---

## Appendix B: Suggested figures

- 1) **State-space sketch:** latent coordination basin vs compensation basin, with arrows showing regulator bias back toward coordination.
- 2) **Time-series plot:**  $\bar{C}(t)$ ,  $H(t)$ , and energy proxy over a perturbation event (with/without regulator).
- 3) **Energy vs duty-cycle:** energy per task vs DC across trials.

**Note on authorship and tools:**

This work was developed through iterative reasoning, modeling, and synthesis. Large language models were used as a collaborative tool to assist with drafting, clarification, and cross-domain translation. All conceptual framing, structure, and final judgments remain the responsibility of the author.