LOCM: BEHAVIORAL REFINEMENT OF REWARD-INDUCED TRAJECTORY EQUIVALENCE IN MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In cooperative multi-agent reinforcement learning, different policies may achieve similar cumulative returns while exhibiting distinct execution behaviors. We interpret this return degeneracy as a structural consequence of compressing high-dimensional joint trajectories into a scalar task return, thereby inducing behaviorally heterogeneous return-equivalent sets. Our theoretical analysis indicates that scalar task returns can locally induce high-dimensional sets of return-equivalent trajectories, while appropriately chosen trajectory-level behavioral representations can further distinguish behavioral variations that remain unresolved by the original task return. Based on this view, we propose LOCM (LLM-Calibrated Observable Coordination Modeling), a trajectory-aware behavioral refinement framework that further refines execution behavior within an ε-optimal task-performance set rather than replacing the original task objective. LOCM combines interpretable joint-action behavioral coordinates, phase-adaptive weighting through an LLM-assisted semantic interface and numerical adapter, and an outcome anchor that constrains task-performance degradation. Experiments on power distribution network control and traffic-control tasks show persistent behavioral differences among near-return trajectories, identify locally reducible execution overhead, and demonstrate that LOCM reduces action variation across multiple MARL backbones while maintaining competitive task performance. These results support task-preserving behavioral refinement as a principled mechanism for mitigating reward-induced execution ambiguity.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) learns joint policies by optimizing shared task rewards (Lowe et al., 2017; Foerster et al., 2018; Rashid et al., 2018; Yu et al., 2022). Yet a scalar cumulative return evaluates task outcome without uniquely specifying how that outcome is produced. Even under matched initial conditions and exogenous inputs, different policies may achieve similar returns while exhibiting substantially different execution processes, such as stronger action-sign counteraction, more frequent dominant-agent switching, or larger temporal action variation. When such differences do not yield commensurate gains in task performance, the task return can indicate that multiple trajectories are similarly successful but provides no further criterion for distinguishing their execution.

This ambiguity is related to, but distinct from, conventional credit assignment (Foerster et al., 2018; Rashid et al., 2018; Son et al., 2019; Zhao and Xie, 2025). Credit assignment asks how a shared return should be attributed to individual agents, actions, or decisions, whereas our question arises after task-level performance is already comparable. We formalize this residual ambiguity as reward-induced trajectory equivalence under the scalar task-return mapping. Exact equality of return induces return-equivalent sets, while empirical near-return matching provides an approximate counterpart for finite-sample analysis. We refer to persistent behavioral heterogeneity within exact return fibers or among matched near-return trajectories as return degeneracy. Under standard local regularity conditions, a scalar task return can leave a high-dimensional family of trajectory variations unresolved, while trajectory-level behavioral representations can further distinguish independent variations within these return-null directions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The goal is not to impose a universal notion of a "good" trajectory. Lower variation, weaker counteraction, or fewer switches are not assumed to be intrinsically preferable under every operating condition. Instead, task performance remains primary, while execution behavior is refined only within a task-acceptable region. We therefore represent joint trajectories through observable behavioral coordinates, including action-sign counteraction, dominant-agent switching, and temporal action variation, which characterize complementary execution differences rather than a single monotonic coordination score. Related work on behavioral diversity similarly shows that distinct policies or skills can coexist under comparable task-quality requirements (Eysenbach et al., 2019; Parker-Holder et al., 2020; Zhou et al., 2022; Zahavy et al., 2023).

Based on this view, we propose LOCM (LLM-Calibrated Observable Coordination Modeling), a trajectory-aware framework for task-preserving behavioral refinement under return degeneracy. LOCM treats behavioral refinement as a secondary preference over a task-acceptable policy region, motivated by an ideal ε-optimal task-performance formulation rather than replacing the original task objective. In practice, task preservation is operationalized relative to an attainable reference performance. At each training phase, an LLM-assisted semantic interface interprets aggregated task and behavioral statistics, and a trainable numerical adapter produces phase-level calibration parameters under semantic supervision. The LLM neither generates real-time actions nor directly specifies numerical reward coefficients. A reference-relative outcome anchor constrains task-performance degradation, keeping behavioral refinement subordinate to the original task objective.

We evaluate LOCM through three stages. Near-return matching examines whether task-comparable trajectories remain behaviorally heterogeneous; controlled open-loop intervention tests whether selected execution activity can be removed without degrading the original environmental return; and end-to-end evaluation assesses whether LOCM reduces execution variation while maintaining competitive task performance. Experiments span multiple MARL backbones in continuous-action power distribution network control and discrete-action traffic-control tasks.

Our contributions are threefold: 1) we formulate return degeneracy as behavioral non-identifiability induced by a scalar task-return mapping and characterize its local geometry; 2) we establish a trajectory-level behavioral refinement principle and instantiate it through observable behavioral coordinates, phase-adaptive LLM-assisted calibration, a numerical adapter, and a reference-relative outcome anchor; and 3) we validate the existence, local reducibility, and end-to-end refinement of reward-induced execution ambiguity through near-return matching, controlled intervention, cross-seed evaluation, multiple MARL backbones, and two control domains.

2 Related Work

Reward shaping and multi-objective or constrained RL. Reward shaping modifies learning signals (Ng et al., 1999; Devlin and Kudenko, 2012), while constrained RL formalizes task requirements through explicit constraints (Achiam et al., 2017; Tessler et al., 2019) and multi-objective RL studies trade-offs among multiple objectives (Roijers et al., 2013; Hayes et al., 2022; Huang et al., 2022; Xue et al., 2025). Diversity-oriented RL further seeks multiple behaviorally distinct yet high-performing policies or skills (Eysenbach et al., 2019; Parker-Holder et al., 2020; Zhou et al., 2022; Zahavy et al., 2023). LOCM addresses a different setting: the original scalar task objective already assigns comparable value to behaviorally different trajectories. Rather than replacing the primary objective, LOCM introduces behavioral information as a secondary refinement criterion within a task-acceptable region.

Cooperation, roles, and credit assignment in MARL. Cooperative MARL includes centralized-training/decentralized-execution and value-decomposition methods (Lowe et al., 2017; Sunehag et al., 2018; Foerster et al., 2018; Rashid et al., 2018; Son et al., 2019; J. Wang et al., 2021), exploration-oriented methods (Mahajan et al., 2019), and on-policy optimization (Yu et al., 2022; Kuba et al., 2022). Other work studies latent roles and behavioral differentiation (T. Wang et al., 2020; 2021; Li et al., 2021; Hu et al., 2024; Dou et al., 2024), inter-agent influence (Jaques et al., 2019), and shared-reward credit assignment (Foerster et al., 2018; Zhao and Xie, 2025), often on cooperative benchmarks such as SMAC (Samvelyan et al., 2019). Credit assignment concerns attribution under the task objective; LOCM instead addresses residual trajectory-level ambiguity after task performance is already comparable. The two are complementary: one attributes the primary objective, whereas the other refines executions that the scalar objective leaves unresolved.

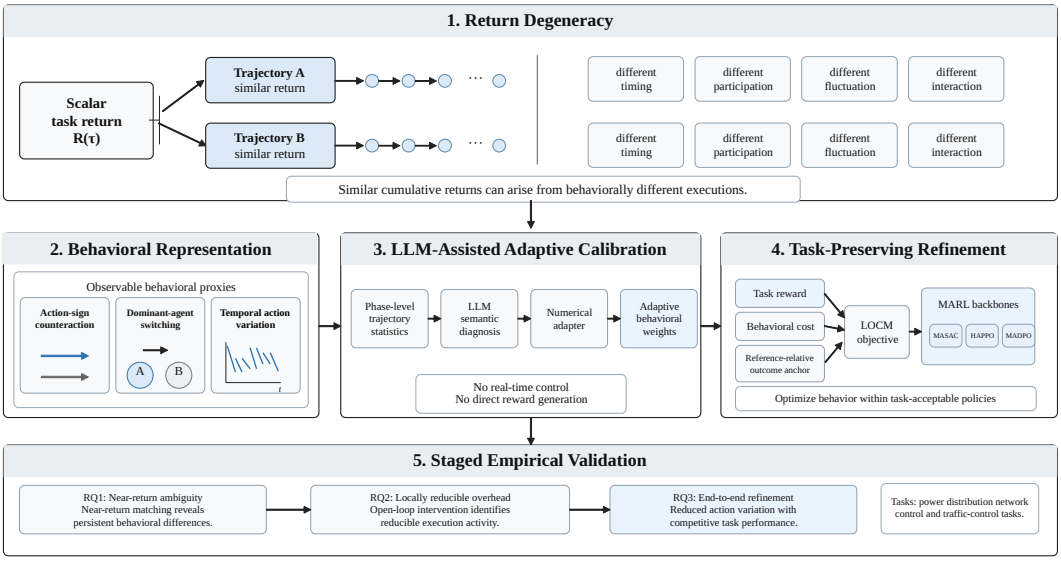


Figure 1: Overview of the proposed LOCM framework for behavioral refinement under return degeneracy.

Language-model-assisted reward design. LLM-based methods such as Language to Rewards, Text2Reward, Eureka, Motif, and adaptive reward-design approaches use language models to generate, evaluate, or refine reward signals and specifications (Yu et al., 2023; Xie et al., 2024; Ma et al., 2024; Klissarov et al., 2024; Sun et al., 2025). LOCM assigns the LLM a narrower role: the task objective and behavioral representation remain predefined, while the LLM performs phase-level semantic diagnosis and a trainable numerical adapter produces the corresponding calibration parameters. Thus, LOCM adapts a secondary behavioral preference rather than redefining the primary task reward.

3 Method: Behavioral Refinement of Reward-Induced Equivalence

3.1 Reward-Induced Trajectory Equivalence

Consider a cooperative multi-agent Markov game with N agents. A joint policy π induces a trajectory $\tau = (s_0, u_0, r_0, \ldots, s_T) \in \mathcal{T}$, where $u_t = [u_t^{(1)}, \ldots, u_t^{(N)}]$ denotes the executed control vector. In continuous-control tasks, u_t is the executed action; in discrete traffic-control tasks, categorical policy outputs are decoded into physical control increments before behavioral analysis.

To analyze reward-induced trajectory equivalence through the task-return mapping, we use the undiscounted episodic environmental return

$$G(\tau) = \sum_{t=0}^{T-1} r_t. \tag{1}$$

Although a trajectory contains high-dimensional states, controls, and interactions, $G(\tau)$ is scalar. For $g \in \mathbb{R}$, define the exact return fiber

$$\mathcal{F}(g) = \{\tau \in \mathcal{T} : G(\tau) = g\}. \tag{2}$$

while empirical near-return pairs satisfy

$$d_G(G(\tau_i), G(\tau_j)) \le \varepsilon_G, \tag{3}$$

where $d_G(a, b) = |a - b| / \max\big((|a| + |b|)/2, 10^{-8}\big)$ is the symmetric relative cumulative-return gap used in RQ1. We call persistent behavioral heterogeneity within exact return fibers or among near-return pairs *return degeneracy*.

For local analysis, suppose a neighborhood $\mathcal{U}$ of feasible trajectories admits a smooth coordinate chart $\phi : \mathcal{U} \to \mathcal{M} \subseteq \mathbb{R}^d$, that is a diffeomorphism onto its image, with local return representation $\widetilde{G}(\phi(\tau)) = G(\tau)$. This analytical construction also applies to discrete-control trajectories through a smooth local embedding or relaxation.

Proposition 1 (Reward-induced local equivalence). If $\widetilde{G}$ is continuously differentiable and $D\widetilde{G}(z_0) \neq 0$, then the local iso-return set $\mathcal{L}_{z_0} = \{z : \widetilde{G}(z) = \widetilde{G}(z_0)\}$ is locally a $(d-1)$-dimensional submanifold satisfying

$$T_{z_0}\mathcal{L}_{z_0} = \ker D\widetilde{G}(z_0). \tag{4}$$

Thus, at regular points, a scalar task return leaves a $(d-1)$-dimensional family of local trajectory variations unresolved, with no first-order discrimination along return-null directions. Appendix A.1 provides the proof.

3.2 Behavioral Refinement of Return Equivalence

To distinguish execution variations unresolved by the task return, we introduce a locally differentiable behavioral representation

$$\Phi(z) = [\Psi_1(z), \ldots, \Psi_Q(z)]^\top. \tag{5}$$

Its independent information within the return-null space is quantified by

$$\rho(z) = \operatorname{rank}\Big(D\Phi(z)\big|_{\ker D\widetilde{G}(z)}\Big). \tag{6}$$

Let $H(z) = [\widetilde{G}(z), \Phi(z)^\top]^\top$.

Theorem 1 (Behavioral refinement of return equivalence). At a regular point z_0, if DH has locally constant rank,

$$\operatorname{rank} DH(z_0) = 1 + \rho(z_0), \qquad \dim H^{-1}(H(z_0)) = d - 1 - \rho(z_0). \tag{7}$$

Hence, behavioral information reduces the local unresolved dimension according to the number of independent return-null directions it reveals. Equation (7) is a representation-level result and does not imply that a single scalarized training signal simultaneously exploits all $\rho(z)$ directions. Appendix A.2 gives the derivation, and Appendix A.4 analyzes scalar behavioral projections.

For the control tasks considered, LOCM uses three statistics of the executed control trajectory:

$$\begin{aligned}
\Psi_{\mathrm{ctr}}(\tau) &= \frac{1}{T}\sum_{t=0}^{T-1} \frac{\|u_t\|_1 - \left|\mathbf{1}^\top u_t\right|}{\|u_t\|_1 + \epsilon_u}, \\
\Psi_{\mathrm{sw}}(\tau) &= \frac{1}{T-1}\sum_{t=1}^{T-1} \mathbb{I}\big[i_t^\star \neq i_{t-1}^\star\big], \qquad i_t^\star = \arg\max_i |u_t^{(i)}|, \\
\Psi_{\mathrm{vol}}(\tau) &= \frac{1}{T-1}\sum_{t=1}^{T-1} \|u_t - u_{t-1}\|_1.
\end{aligned} \tag{8}$$

We denote $\Phi_{\mathrm{obs}}(\tau) = [\Psi_{\mathrm{ctr}}(\tau), \Psi_{\mathrm{sw}}(\tau), \Psi_{\mathrm{vol}}(\tau)]^\top$. These quantities are observable behavioral proxies, not sufficient statistics or universally monotonic measures of coordination quality. For traffic control, they are computed from decoded physical control increments rather than categorical action labels. Appendix A.3 provides smooth relaxation for the differential analysis.

3.3 Task-Anchored Secondary Optimization

Behavioral refinement should resolve execution ambiguity without replacing the original task objective. Let $J_{\mathrm{eval}}(\pi) = \mathbb{E}_{\tau\sim p_\pi}[G(\tau)]$, $J^\star = \sup_{\pi\in\Pi} J_{\mathrm{eval}}(\pi)$, and define the ε-optimal task-performance set

$$\Pi_\varepsilon = \{\pi : J_{\mathrm{eval}}(\pi) \geq J^\star - \varepsilon\}, \qquad \varepsilon \geq 0. \tag{9}$$

At phase k, let $w^{(k)} \in \mathbb{R}_+^Q$ denote nonnegative phase-dependent behavioral calibration weights, and define

$$C_k^{\mathrm{obs}}(\tau) = w^{(k)\top}\Phi_{\mathrm{obs}}(\tau), \qquad J_{\mathrm{coord}}^{(k)}(\pi) = \mathbb{E}[C_k^{\mathrm{obs}}(\tau)]. \tag{10}$$

The ideal behavioral-refinement principle is

$$\pi_k^\star \in \arg\min_{\pi\in\Pi_\varepsilon} J_{\mathrm{coord}}^{(k)}(\pi). \tag{11}$$

Thus, task performance defines the admissible policy region, while behavioral criteria act only as secondary preferences. The ε-optimal set defines an ideal task-prioritization principle; the practical LOCM implementation does not perform exact projection onto this set, but operationalizes the same principle through reference-relative task preservation.

Because $J^\star$ is unavailable during training, LOCM uses an attainable reference performance J_{ref} and defines the evaluation-level violation

$$V_\delta(\pi) = [J_{\mathrm{ref}} - \delta - J_{\mathrm{eval}}(\pi)]_+, \tag{12}$$

where δ is the allowed reference-relative degradation. This measure links the ideal admissible-set principle in Eq. (11) to the optimization-level relaxation in Section 3.4. The practical realization is implementation dependent; our power-distribution experiments use the reference-relative physical acceptance gate in Appendix F.3.2.

3.4 Phase-Adaptive Behavioral Calibration and Learning

The behavioral representation Φ may expose multiple execution variations unresolved by the scalar task return, whereas a MARL optimizer ultimately requires a scalar training signal. LOCM therefore uses phase-adaptive calibration to determine which behavioral coordinates should be emphasized as the policy evolves.

At phase k, a trajectory batch $\mathcal{B}_k$ is summarized as $s_k = \mathcal{S}(\mathcal{B}_k)$, where s_k contains task-performance and behavioral statistics. An LLM-assisted semantic interface interprets the current behavioral regime,

$$p_k = P_{\mathrm{LLM}}(s_k), \qquad y_k = h(p_k), \tag{13}$$

where p_k is the structured semantic output and y_k is its numerical supervision target. Let x_k denote the adapter feature vector containing current phase statistics, reference-relative physical ratios, and the previous control state. The trainable numerical adapter then produces

$$c_k = A_\phi(x_k). \tag{14}$$

The behavioral-calibration component c_k^{beh} is normalized to obtain the phase-dependent weights

$$w^{(k)} = \mathcal{N}(c_k^{\mathrm{beh}}), \qquad C_k(\tau) = w^{(k)\top}\Phi(\tau). \tag{15}$$

Equation (15) provides the normalized analytical representation used in the formulation above. In the implemented MASAC-LOCM training path, however, the first three bounded adapter outputs are used directly as behavioral penalty coefficients rather than being simplex-normalized; the implementation is detailed in Appendix F.3.1.

Counteraction admits a direct step-level realization, whereas dominant-agent switching and temporal action variation depend on temporal context. When exact decomposition is unavailable, LOCM uses the corresponding step-wise or episode-level surrogate realization. The remaining two components are auxiliary implementation variables governing phase-level action intensity and collective pull. They are not part of the behavioral representation in Eq. (8), the geometric analysis in Sections 3.1–3.2, or the behavioral diagnostics used in RQ1–RQ2. The LLM is invoked only at phase boundaries and does not directly generate rewards or real-time actions; policy optimization remains with the native MARL backbone.

For analysis, we use the scalar penalized surrogate

$$r_t'^{(k)} = r_{\mathrm{task},t} - \beta c_t^{(k)} - \lambda\,\mathrm{av}_t^{(k)}. \tag{16}$$

Table 1: MASAC–MAPPO matching results and behavioral differences at three return tolerances.

Statistic	Relative gap $\leq 2\%$	Relative gap $\leq 3\%$	Relative gap $\leq 5\%$
Raw matched pairs	345	3,210	7,410
Deduplicated pairs	23	214	494
Scenario coverage	12/50	50/50	50/50
Absolute cumulative-return gap	0.224 [0.213, 0.236]	0.300 [0.270, 0.328]	0.394 [0.307, 0.501]
Absolute mean-action-change gap	0.315 [0.314, 0.325]	0.315 [0.310, 0.324]	0.315 [0.310, 0.321]
Absolute effective-counteraction-rate gap	0.372 [0.367, 0.377]	0.379 [0.372, 0.396]	0.389 [0.372, 0.399]
Burden-allocation L_1 gap	0.550 [0.549, 0.552]	0.552 [0.549, 0.557]	0.551 [0.548, 0.558]
Joint executed-action RMSE	0.361 [0.358, 0.362]	0.359 [0.357, 0.362]	0.357 [0.354, 0.359]

This optimization-level surrogate, analyzed in Appendices B–C, is not assumed to be numerically identical to the evaluation-level criterion in Eq. (11). Each backbone retains its native optimizer, while the power-distribution implementation uses the physical acceptance gate in Appendix F.3.2. For the implemented MASAC-LOCM guided phase, Eq. (16) is an analytical surrogate rather than the literal training-reward formula: the raw environmental reward is retained for task evaluation and logging but is not directly added to the guided-phase training reward. The corresponding mixed temporal realization is detailed in Appendix F.1.

4 Experiments

We organize the evaluation around three questions: RQ1, whether near-return trajectories remain behaviorally heterogeneous; RQ2, whether the behavioral representation can identify locally reducible execution activity in settings with nontrivial control activity; and RQ3, whether LOCM reduces execution variation while maintaining competitive task performance across backbones and domains.

4.1 Experimental Setup

RQ1–RQ3 primarily use power distribution network control. RQ1 evaluates 15 late-training MASAC checkpoints (SAC-based; Haarnoja et al., 2018) and 15 late-training MAPPO checkpoints (Yu et al., 2022) over 50 matched scenarios; RQ2 performs controlled open-loop interventions on the resulting near-return trajectories; and RQ3 evaluates MASAC, HAPPO (Kuba et al., 2022), MADPO (Dou et al., 2024), and their LOCM variants under standard and monthly protocols. A discrete-action traffic-control task provides cross-domain validation. Experimental and implementation details are provided in Appendix E, with additional reproducibility materials in Appendix F.

4.2 RQ1: Does Return Degeneracy Exist?

Trajectories are matched using only cumulative environmental return; behavioral quantities are excluded from pair selection. RQ1 intentionally reports a broader set of post-matching execution descriptors that is not restricted to Φ_{obs}; these descriptors are used only to characterize residual behavioral heterogeneity after return-based matching. Raw matches are deduplicated by trajectory-output equivalence within each algorithm and scenario, with the detailed equivalence rule reported in Appendix E.6. At 2% tolerance, 345 raw matches reduce to 23 unique pairs over 12/50 scenarios. Despite a median return gap of only 0.224, substantial differences remain in action change, counteraction, burden allocation, and joint-action RMSE. The pattern persists at 3% and 5%.

Thus, cumulative-return proximity does not uniquely determine execution behavior. A within-MASAC checkpoint analysis shows the same qualitative pattern, indicating that the observed behavioral heterogeneity is not solely attributable to cross-algorithm differences (Appendix D.1).

4.3 RQ2: Can the Behavioral Representation Identify Locally Reducible Execution Activity?

Behavioral heterogeneity alone does not imply inefficiency. We therefore use the observable behavioral representation Φ_{obs} as a diagnostic and intervene on saved near-return trajectories. We call execution activity locally reducible execution overhead when its removal does not degrade the

Table 2: Equal-budget interventions under a 10% realized action-removal budget.

Algorithm	Selection rule	Unique trajectories	Return non-decrease	Median ΔG	Mean ΔG $\pm$ SD	Median actual removal	Median Δ voltage deviation	Median Δ loss energy
MASAC	Φ-based diagnostic	23	100%	+0.264	$+0.266 \pm 0.011$	8.425	-0.737	-0.115
MASAC	Action magnitude	23	100%	+0.247	$+0.206 \pm 0.056$	8.425	-0.449	-0.133
MASAC	Action change	23	100%	+0.144	$+0.128 \pm 0.034$	8.425	-0.581	-0.126
MASAC	Random (3 repeats)	23	72.5%	+0.148	$+0.117 \pm 0.205$	8.425	-0.011	-0.087
MAPPO	Φ-based diagnostic	12	0%	-0.0001	-0.0001 ± 0.0000	0.023	+0.0002	-0.0001
MAPPO	Action magnitude	12	100%	+0.0010	$+0.0011 \pm 0.0000$	0.023	-0.0017	-0.0005
MAPPO	Action change	12	0%	-0.0005	-0.0005 ± 0.0001	0.023	+0.0008	-0.0002
MAPPO	Random (3 repeats)	12	36.1%	-0.0003	-0.0005 ± 0.0011	0.023	+0.0005	-0.0003

Table 3: Performance under the standard MEDIUM protocol.

Method	Return G	Line-loss	Action mag.	Action variation	Counteraction rate	Role-switch rate
MASAC	-12.059	1.798	0.192	0.211	0.656	0.537
MASAC-LOCM	-11.559	1.252	0.048	0.040	0.719	0.463
HAPPO	-11.132	2.013	0.307	0.332	0.646	0.505
HAPPO-LOCM	-11.185	2.002	0.258	0.268	0.521	0.421
MADPO	-10.784	1.982	0.301	0.338	0.635	0.484
MADPO-LOCM	-11.145	1.935	0.246	0.259	0.490	0.474

original environmental return under the matched open-loop intervention. The resulting Φ-based diagnostic is compared with action magnitude, action change, and random selection under identical initial conditions and exogenous inputs. This experiment evaluates whether the behavioral representation can identify locally reducible activity under matched open-loop interventions, rather than the full training-time LOCM objective.

4.3.1 Equal-Budget Intervention

For each trajectory, the removal budget is $B = 0.10 \sum_t \|u_t\|_1$. Time steps are processed in descending selection-score order, with the final selected step uniformly scaled when necessary so that the removed L_1 action magnitude matches B up to numerical tolerance.

Under this matched budget, the Φ-based diagnostic yields the largest MASAC mean and median return gains (+0.266 and +0.264) and the largest reduction in cumulative voltage deviation, while MAPPO effects remain near zero. Additional results are reported in Appendices D.2–D.3.

4.4 RQ3: Can LOCM Reduce Execution Variation while Maintaining Competitive Task Performance?

RQ3 combines checkpoint-level evaluation with independent-seed training. Standard and monthly protocols compare MASAC, HAPPO, MADPO, and their LOCM variants, while HAPPO and MADPO are additionally retrained from independent seeds to assess the cross-seed reproducibility of the behavioral-refinement effect.

4.4.1 Power Distribution Network Control

Under the standard MEDIUM protocol, all three LOCM variants reduce action magnitude, temporal variation, and line loss. MASAC-LOCM also improves return, whereas HAPPO-LOCM and MADPO-LOCM exhibit modest return decreases, revealing a backbone-dependent task–behavior trade-off. Full metrics are reported in Appendix D.4.

Colors denote MARL backbones; solid-circle and dashed-square curves denote baseline and LOCM variants, respectively. Panels show (a) line-loss energy, (b) maximum voltage peak, (c) minimum voltage peak, (d) cumulative voltage deviation, and (e) counteraction rate.

Across the 31-day monthly sequence, LOCM generally reduces line-loss energy and shifts the voltage extrema toward the nominal range across the three MARL backbones. HAPPO-LOCM and MADPO-LOCM also reduce cumulative voltage deviation over most operating days, whereas MASAC-LOCM exhibits a clear increase, revealing a backbone-dependent physical trade-off.

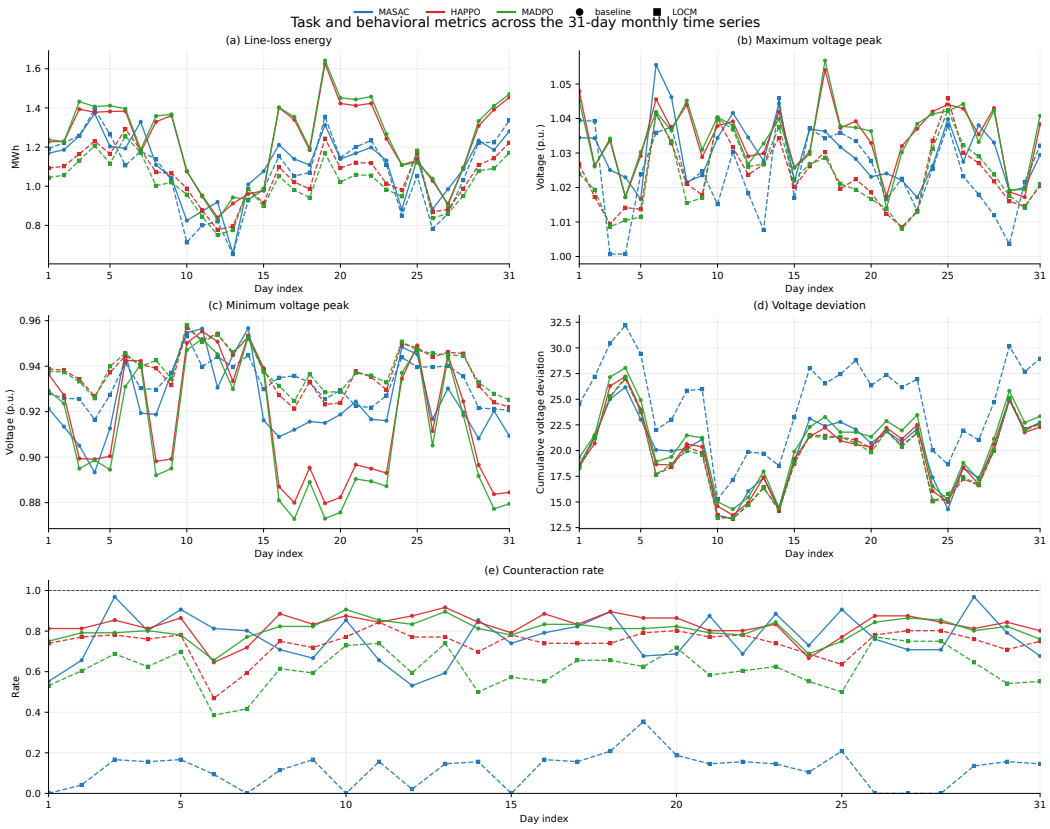


Figure 2: Physical and behavioral performance across the 31-day monthly time series.

Table 4: Cross-seed evaluation of HAPPO and MADPO (mean ± SD).

Method	Return G	Voltage dev.	Action mag.	Action variation
HAPPO	-12.6107 ± 0.0812	0.2124 ± 0.0052	0.1505 ± 0.0552	0.0437 ± 0.0276
HAPPO-LOCM	-12.4348 ± 0.4225	0.2122 ± 0.0096	0.1309 ± 0.0117	0.0346 ± 0.0095
MADPO	-12.5793 ± 0.3200	0.2143 ± 0.0050	0.1430 ± 0.0493	0.0415 ± 0.0255
MADPO-LOCM	-12.5586 ± 0.5271	0.2140 ± 0.0111	0.1289 ± 0.0025	0.0362 ± 0.0073

Counteraction is reduced by all three LOCM variants, with the largest reduction observed for MASAC. Overall, the monthly results show that LOCM alters physical and behavioral execution characteristics under time-varying operating conditions, with generally lower line-loss energy and counteraction but a backbone-dependent effect on cumulative voltage deviation.

4.4.2 Independent Training-Seed Analysis

Across three independent training seeds (3, 4, and 5), LOCM consistently reduces action magnitude and temporal variation for both HAPPO and MADPO while maintaining comparable task performance. Evaluation metrics are first aggregated within each trained model and then reported as mean ± SD across training seeds. Mean action variation decreases by 20.8% for HAPPO and 12.8% for MADPO, with concurrent reductions in action magnitude. These results support cross-seed reproducibility of the behavioral-refinement effect rather than improved training stability.

4.4.3 Cross-Domain Traffic-Control Validation

We further evaluate LOCM on discrete-action traffic signal control under unseen traffic sequences. Across low-, medium-, and high-demand traffic sequences, LOCM reduces average delay, maximum queue occupancy, action variation, action magnitude, and counteraction while increasing through-

Table 5: Traffic signal control under unseen traffic sequences (mean ± SD).

Metric	Low SAC	Low LOCM	Medium SAC	Medium LOCM	High SAC	High LOCM
Avg. delay	0.457 ± 0.010	0.374 ± 0.005	0.570 ± 0.003	0.486 ± 0.0033	0.689 ± 0.014	0.591 ± 0.004
Queue overflow	2.432 ± 0.386	2.321 ± 0.250	4.347 ± 0.342	4.068 ± 0.308	6.178 ± 0.349	5.755 ± 0.411
Throughput	72.519 ± 0.28	72.921 ± 0.43	69.405 ± 0.28	69.849 ± 0.336	66.15 ± 0.195	66.58 ± 0.244
Max. queue	0.600 ± 0.007	0.572 ± 0.006	0.654 ± 0.003	0.641 ± 0.0025	0.696 ± 0.005	0.686 ± 0.007
Action variation	0.091 ± 0.006	0.026 ± 0.002	0.091 ± 0.006	0.032 ± 0.0031	0.092 ± 0.0028	0.036 ± 0.005
Action mag.	0.059 ± 0.003	0.017 ± 0.001	0.060 ± 0.003	0.021 ± 0.0014	0.061 ± 0.0024	0.025 ± 0.003
Counteraction	0.576 ± 0.022	0.069 ± 0.016	0.549 ± 0.034	0.090 ± 0.043	0.590 ± 0.053	0.139 ± 0.052

Table 6: Compact ablation of major LOCM components.

Variant	Voltage dev.	Network loss	Action mag.	Action variation
Rule-diagnoser	0.228 ± 0.012	0.073 ± 0.0025	0.087 ± 0.0012	0.095 ± 0.0024
No-coord	0.259 ± 0.0076	0.168 ± 0.0047	0.246 ± 0.0043	0.328 ± 0.011
Static-coord	0.220 ± 0.0092	0.073 ± 0.0010	0.093 ± 0.0028	0.096 ± 0.0025
No-adapter	0.212 ± 0.0058	0.074 ± 0.0009	0.087 ± 0.0040	0.091 ± 0.0019
No-anchor	0.215 ± 0.0070	0.074 ± 0.0017	0.089 ± 0.0013	0.093 ± 0.0018
LOCM	0.197 ± 0.0019	0.054 ± 0.0003	0.047 ± 0.0003	0.042 ± 0.0029

put. Additional behavioral metrics reported in Appendix D.7 show increased role switching and burden trend, indicating that the evaluated behavioral metrics reflect distinct execution characteristics rather than a single monotonic notion of coordination quality.

4.5 Ablation Analysis

Ablations reveal complementary rather than uniformly monotonic component effects. Removing behavioral refinement causes marked degradation in task-level physical metrics, action magnitude, and action variation, while static weighting, rule-based diagnosis, removing the numerical adapter, and removing the outcome anchor affect different subsets of task and behavioral criteria. Full component-level results corresponding to Table 6 are reported in Appendix D.6.

Additional comparisons with fixed, rule-based, and semantically shuffled calibration are provided in Appendix D.8, showing that LLM-assisted calibration provides a distinct and competitive task–behavior trade-off while preserving effective trajectory-level behavioral refinement.

Together, RQ1–RQ3 show that near-return trajectories remain behaviorally heterogeneous, the behavioral representation can identify locally reducible execution activity in settings with nontrivial control activity, and LOCM reduces action variation across backbones and domains while maintaining competitive task performance.

5 Conclusion

We study reward-induced trajectory equivalence in cooperative MARL, where behaviorally distinct trajectories can remain unresolved by a scalar task-return mapping. We show that trajectory-level behavioral representations can further distinguish this behavioral non-identifiability along return-null directions, and propose LOCM to realize task-preserving behavioral refinement through observable executed-control statistics, phase-adaptive calibration, and a reference-relative outcome anchor. Experiments across near-return matching, controlled interventions, multiple MARL backbones, and two control domains reveal persistent behavioral differences among task-comparable trajectories, provide evidence of locally reducible execution activity, and show that LOCM can reduce execution variation while maintaining competitive task performance. Additional calibration-strategy comparisons indicate that LLM-assisted phase adaptation selects a distinct task–behavior operating point, while the underlying behavioral-refinement mechanism remains effective under alternative fixed, rule-based, or semantically shuffled calibration strategies. The current theory is local, and the empirical conclusions are limited to the evaluated behavioral proxies, control domains, calibration mechanisms, and training settings.

AI USE STATEMENT

Generative AI tools were used in this work in two roles. First, a large language model is an explicit component of the proposed LOCM framework, where it is used only for phase-level semantic diagnosis and does not generate real-time control actions or directly specify numerical reward coefficients; this use is described in Sections 3.4, E.5, and F.2–F.3. Second, generative AI tools were used during manuscript preparation for language editing, LaTeX formatting, and consistency checking. All AI-assisted text, mathematical expressions, citations, and formatting were reviewed by the authors. The experimental design, reported numerical results, figures, and scientific conclusions were verified by the authors, who take full responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

We provide detailed materials to support reproducibility throughout the paper and appendices. The theoretical assumptions, proofs, and supplementary derivations are given in Appendices A–C. Additional experimental results and diagnostic analyses are reported in Appendix D. Environment configurations, MARL backbone hyperparameters, LOCM-specific calibration settings, the LLM interface, and evaluation protocols are documented in Appendix E. Appendix F further specifies the phase-adaptive training procedure, semantic prompt structure, structured LLM output, numerical adaptation procedure, reference-relative outcome anchoring, and fallback mechanism. Independent training-seed evaluations and matched evaluation protocols are reported where applicable to distinguish variation across trained models from variation across operating scenarios.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31, 2017.

Sam Devlin and Daniel Kudenko. Dynamic potential-based reward shaping. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, pp. 433–440, 2012.

Haowen Dou, Lujuan Dang, Zhirong Luan, and Badong Chen. Measuring mutual policy divergence for multi-agent sequential exploration. In *Advances in Neural Information Processing Systems*, volume 37, pp. 76265–76288, 2024. doi: 10.52202/079017-2428.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019.

Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 2974–2982, 2018. doi: 10.1609/aaai.v32i1.11794.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, 2018.

Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36:26, 2022. doi: 10.1007/s10458-022-09552-y.

Zican Hu, Zongzhang Zhang, Huaxiong Li, Chunlin Chen, Hongyu Ding, and Zhi Wang. Attention-guided contrastive role representations for multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2024.

Sandy Huang, Abbas Abdolmaleki, Giulia Vezzani, Philemon Brakel, Daniel J. Mankowitz, Michael Neunert, Steven Bohez, Yuval Tassa, Nicolas Heess, Martin Riedmiller, and Raia Hadsell. A constrained multi-objective reinforcement learning framework. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 883–893, 2022.

Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, Dj Strouse, Joel Z. Leibo, and Nando de Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3040–3049, 2019.

Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. In *International Conference on Learning Representations*, 2024.

Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2022.

John M. Lee. *Introduction to Smooth Manifolds*. Volume 218 of *Graduate Texts in Mathematics*. Springer, second edition, 2012. doi: 10.1007/978-1-4419-9982-5.

Chenghao Li, Tonghan Wang, Chengjie Wu, Qianchuan Zhao, Jun Yang, and Chongjie Zhang. Celebrating diversity in shared multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations*, 2024.

Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. MAVEN: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287, 1999.

Jack Parker-Holder, Aldo Pacchiano, Krzysztof M. Choromanski, and Stephen J. Roberts. Effective diversity in population based reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4295–4304, 2018.

Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013. doi: 10.1613/JAIR.3987.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5887–5896, 2019.

Shengjie Sun, Runze Liu, Jiafei Lyu, Jing-Wen Yang, Liangpeng Zhang, and Xiu Li. A large language model-driven reward design framework via dynamic feedback for reinforcement learning. *Knowledge-Based Systems*, 326:114065, 2025. doi: 10.1016/j.knosys.2025.114065.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex dueling multi-agent Q-learning. In *International Conference on Learning Representations*, 2021.

Tonghan Wang, Heng Dong, Victor Lesser, and Chongjie Zhang. ROMA: Multi-agent reinforcement learning with emergent roles. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9876–9886, 2020.

Tonghan Wang, Tarun Gupta, Bei Peng, Anuj Mahajan, Shimon Whiteson, and Chongjie Zhang. RODE: Learning roles to decompose multi-agent tasks. In *International Conference on Learning Representations*, 2021.

Tianbao Xie, Siheng Zhao, Chen Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2Reward: Reward shaping with language models for reinforcement learning. In *International Conference on Learning Representations*, 2024.

Bo Xue, Dake Bu, Ji Cheng, Yuanyu Wan, and Qingfu Zhang. Multi-objective linear reinforcement learning with lexicographic rewards. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70065–70085, 2025.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montserrat Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 374–404, 2023.

Tom Zahavy, Yannick Schroecker, Feryal M. P. Behbahani, Kate Baumli, Sebastian Flennerhag, Shaobo Hou, and Satinder Singh. Discovering policies with DOMiNO: Diversity optimization maintaining near optimality. In *International Conference on Learning Representations*, 2023.

Xutong Zhao and Yaqi Xie. Multi-level advantage credit assignment for cooperative multi-agent reinforcement learning. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 2971–2979, 2025.

Zihan Zhou, Wei Fu, Bingliang Zhang, and Yi Wu. Continuously discovering novel strategies via reward-switching policy optimization. In *International Conference on Learning Representations*, 2022.

A Geometric Analysis of Reward-Induced Equivalence

A.1 Proof of Proposition 1

Let $\mathcal{M}$ be a d-dimensional differentiable manifold representing a local neighborhood of feasible trajectories, and let

$$\widetilde{G} : \mathcal{M} \to \mathbb{R}$$

be continuously differentiable in a neighborhood of $z_0 \in \mathcal{M}$. By assumption,

$$D\widetilde{G}(z_0) \neq 0,$$

so the differential of $\widetilde{G}$ has rank one at z_0.

Let

$$g_0 = \widetilde{G}(z_0)$$

and define the local iso-return set through z_0 as

$$\mathcal{F}_{\mathrm{loc}}(g_0) = \{z \in \mathcal{M} : \widetilde{G}(z) = g_0\}. \tag{A1}$$

By the regular level-set theorem, $\mathcal{F}_{\mathrm{loc}}(g_0)$ is locally an embedded submanifold of codimension one. Therefore,

$$\dim \mathcal{F}_{\mathrm{loc}}(g_0) = d - 1. \tag{A2}$$

To characterize its tangent space, consider any differentiable curve

$$c : (-\epsilon, \epsilon) \to \mathcal{F}_{\mathrm{loc}}(g_0)$$

satisfying $c(0) = z_0$. Since

$$\widetilde{G}(c(t)) = g_0$$

along the curve, differentiation at $t = 0$ gives

$$\frac{d}{dt}\widetilde{G}(c(t))\bigg|_{t=0} = D\widetilde{G}(z_0)\dot{c}(0) = 0. \tag{A3}$$

Hence,

$$T_{z_0}\mathcal{F}_{\mathrm{loc}}(g_0) \subseteq \ker D\widetilde{G}(z_0). \tag{A4}$$

Conversely, for a regular level set, every vector in $\ker D\widetilde{G}(z_0)$ is tangent to the level set. Thus,

$$T_{z_0}\mathcal{F}_{\mathrm{loc}}(g_0) = \ker D\widetilde{G}(z_0). \tag{A5}$$

Because $D\widetilde{G}(z_0)$ has rank one, its kernel has dimension $d - 1$. Consequently, a scalar task return leaves a $(d-1)$-dimensional family of local trajectory variations unresolved to first order. In particular, for any

$$v \in \ker D\widetilde{G}(z_0),$$

we have

$$D\widetilde{G}(z_0)v = 0,$$

so the scalar return provides no first-order local discrimination along such return-null directions.

This result is local: it characterizes the geometry of the return fiber in a neighborhood of a regular point and does not imply that globally return-equivalent trajectories form a single connected component.

A.2 Projector-Based Derivation of Behavioral Refinement

Let

$$J_G(z) = D\widetilde{G}(z) \in \mathbb{R}^{1\times d}$$

and

$$J_\Phi(z) = D\Phi(z) \in \mathbb{R}^{Q\times d}.$$

At a regular point z_0, $J_G(z_0) \neq 0$. Define the orthogonal projector onto the local return-null subspace as

$$P_G^\perp = I_d - \frac{J_G(z_0)^\top J_G(z_0)}{\|J_G(z_0)\|_2^2}. \tag{A6}$$

The range of $P_G^\perp$ is exactly

$$\mathrm{Range}(P_G^\perp) = \ker J_G(z_0),$$

which, by Proposition 1, coincides with the tangent space of the local iso-return set.

Define

$$B(z_0) = J_\Phi(z_0)P_G^\perp. \tag{A7}$$

The rows of $B(z_0)$ describe the first-order sensitivities of the behavioral representation along directions that leave the task return locally unchanged. Hence,

$$\rho(z_0) = \mathrm{rank}\left(J_\Phi(z_0)\big|_{\ker J_G(z_0)}\right) = \mathrm{rank}\, B(z_0). \tag{A8}$$

Now consider the augmented representation

$$H(z) = \begin{bmatrix}\widetilde{G}(z)\\ \Phi(z)\end{bmatrix},$$

whose Jacobian is

$$DH(z_0) = \begin{bmatrix}J_G(z_0)\\ J_\Phi(z_0)\end{bmatrix}. \tag{A9}$$

Each row $b_j^\top$ of $J_\Phi(z_0)$ can be decomposed into a component parallel to $J_G(z_0)$ and a component in the return-null subspace:

$$b_j^\top = b_{j,\parallel}^\top + b_{j,\perp}^\top. \tag{A10}$$

Subtracting the component parallel to $J_G(z_0)$ from each behavioral row is an elementary row operation and therefore leaves the rank unchanged. The remaining behavioral rows are precisely represented by $J_\Phi(z_0)P_G^\perp$. Therefore,

$$\mathrm{rank}\, DH(z_0) = 1 + \mathrm{rank}\left(J_\Phi(z_0)P_G^\perp\right) = 1 + \rho(z_0). \tag{A11}$$

Under the locally constant-rank assumption, the constant-rank theorem implies that the local fiber of the augmented representation,

$$H^{-1}(H(z_0)),$$

has dimension

$$\dim H^{-1}(H(z_0)) = d - \mathrm{rank}\, DH(z_0) = d - 1 - \rho(z_0). \tag{A12}$$

Thus, relative to the $(d-1)$-dimensional local return fiber induced by the scalar task return, the behavioral representation resolves $\rho(z_0)$ locally independent directions that were previously indistinguishable.

This establishes Theorem 1. □

For additional interpretation, let the nonzero singular values of $B(z_0) = J_\Phi(z_0)P_G^\perp$ be

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_\rho > 0. \tag{A13}$$

For a sufficiently small return-null perturbation $v \in \ker J_G(z_0)$,

$$\Phi(z_0 + v) = \Phi(z_0) + B(z_0)v + o(\|v\|). \tag{A14}$$

Accordingly, $\rho(z_0)$ measures the number of locally independent behavioral directions exposed within the return-null space, while the nonzero singular values of $B(z_0)$ characterize the local sensitivity of the behavioral representation along those directions. This singular-value interpretation is supplementary and does not introduce an additional identifiability claim beyond Theorem 1.

A.3 Smooth Relaxations of Observable Behavioral Proxies

The practical behavioral coordinates in Eq. (8) contain absolute values, an arg max operation, and indicator functions. These quantities are directly observable and are used in all empirical evaluations, but they are not differentiable everywhere. To connect the practical coordinates to the local differential analysis in Sections 3.1–3.2, we introduce smooth approximations solely for theoretical analysis.

For $\eta > 0$, define the smooth absolute-value approximation

$$\varphi_\eta(x) = \sqrt{x^2 + \eta^2}, \tag{A15}$$

and the corresponding smooth L_1-type magnitude

$$\|u\|_{1,\eta} = \sum_i \varphi_\eta(u_i). \tag{A16}$$

Smooth action-sign counteraction. A smooth approximation to the action-sign counteraction statistic is

$$\widetilde{\Psi}_{\mathrm{ctr}} = \frac{1}{T} \sum_{t=0}^{T-1} \frac{\|u_t\|_{1,\eta} - \varphi_\eta(\mathbf{1}^\top u_t)}{\|u_t\|_{1,\eta} + \epsilon_u}. \tag{A17}$$

As $\eta \to 0$, this expression approaches the practical counteraction statistic away from zero-valued nondifferentiable configurations.

Smooth dominant-agent switching. To relax the dominant-agent identity, define soft dominance probabilities

$$p_{t,i} = \frac{\exp\Big(\varphi_\eta(u_t^{(i)})/\tau_s\Big)}{\sum_j \exp\Big(\varphi_\eta(u_t^{(j)})/\tau_s\Big)}, \tag{A18}$$

where $\tau_s > 0$ is a softmax temperature.

A smooth approximation to dominant-agent switching is then

$$\widetilde{\Psi}_{\mathrm{sw}} = \frac{1}{T-1} \sum_{t=1}^{T-1} \left(1 - \sum_i p_{t,i} p_{t-1,i}\right). \tag{A19}$$

As $\tau_s \to 0$, the soft dominance vector approaches a one-hot representation whenever the largest action magnitude is unique. Consequently, Eq. (A19) approaches the hard switching indicator away from dominant-action ties.

Smooth temporal action variation. The temporal-action-variation coordinate is relaxed as

$$\widetilde{\Psi}_{\mathrm{vol}} = \frac{1}{T-1} \sum_{t=1}^{T-1} \sum_i \varphi_\eta\left(u_t^{(i)} - u_{t-1}^{(i)}\right). \tag{A20}$$

As $\eta \to 0$, Eq. (A20) approaches the L_1-based temporal-action-variation statistic used in the empirical evaluation.

Collectively,

$$\widetilde{\Phi}_{\mathrm{obs}} = \begin{bmatrix} \widetilde{\Psi}_{\mathrm{ctr}} \\ \widetilde{\Psi}_{\mathrm{sw}} \\ \widetilde{\Psi}_{\mathrm{vol}} \end{bmatrix} \tag{A21}$$

provides a differentiable local surrogate for the observable behavioral representation.

These smooth relaxations are introduced only to support the differential analysis. All empirical results use the original observable definitions in Eq. (8), rather than the relaxed quantities above. Moreover, convergence of the smooth proxies to their hard counterparts is understood away from action ties and zero-valued nondifferentiable configurations.

A.4 Rank Exposed by Scalar Behavioral Projections

Theorem 1 concerns the vector-valued behavioral representation Φ, whereas practical MARL optimization ultimately requires a scalar learning signal.

At a fixed local point z_0, consider a scalar behavioral projection

$$\widetilde{C}_w(z) = w^\top \Phi(z), \qquad w \in \mathbb{R}^Q. \tag{A22}$$

Its differential is

$$D\widetilde{C}_w(z_0) = w^\top J_\Phi(z_0). \tag{A23}$$

Restricted to the return-null subspace,

$$D\widetilde{C}_w(z_0)P_G^\perp = w^\top J_\Phi(z_0)P_G^\perp = w^\top B(z_0). \tag{A24}$$

Therefore, the rank exposed by a single scalar behavioral projection is

$$\rho_w(z_0) = \operatorname{rank}\left(w^\top B(z_0)\right) \le 1. \tag{A25}$$

Thus, although the full vector representation may expose several independent return-null behavioral directions, a single scalarized behavioral criterion contributes at most one locally independent first-order constraint.

Now consider K scalar projections evaluated at the same fixed local point, $\widetilde{C}_{w_1}, \ldots, \widetilde{C}_{w_K}$, and define the weight matrix

$$W = \begin{bmatrix} w_1^\top \\ \vdots \\ w_K^\top \end{bmatrix} \in \mathbb{R}^{K\times Q}. \tag{A26}$$

The combined return-null behavioral sensitivity is $WB(z_0)$, whose rank satisfies

$$\operatorname{rank}\left(WB(z_0)\right) \le \min\{K, \rho(z_0)\}. \tag{A27}$$

Therefore, at a fixed local point, distinct scalar projections can probe different components of the behavioral information exposed by Φ. With sufficiently diverse projection directions, their combined span can reveal more of the rank-$\rho(z_0)$ return-null behavioral structure than any individual scalarization.

Equation (A27), however, is strictly a fixed-point local characterization. During training, the policy and the induced trajectory representation evolve, so the corresponding points z_k, return-null spaces, and Jacobians generally change across phases. Consequently, ranks associated with different training phases cannot be directly accumulated in a common coordinate system.

Accordingly, phase-varying scalarization motivates broader exploration of the observable behavioral representation, but it does not guarantee recovery of all return-null behavioral directions or realization of the full representation-level rank established in Theorem 1.

B Reference-Relative Outcome-Anchor Analysis

B.1 Finite-Penalty Bound

For the optimization-level analysis of Eq. (16), let J_γ denote the backbone objective and define

$$\min_\pi \; F_k(\pi) + \lambda_a V_\delta^\gamma(\pi), \tag{B1}$$

where

$$F_k(\pi) = -J_\gamma(\pi) + \beta J^k_{\text{coord}}(\pi), \tag{B2}$$

and

$$V^\gamma_\delta(\pi) = \left[J^\gamma_{\text{ref}} - \delta - J_\gamma(\pi)\right]_+ . \tag{B3}$$

Thus, V^γ_δ is distinct from the evaluation-level violation V_δ in Section 3.3.

Let π_λ denote a global minimizer of Eq. (B1). Assume that F_k has bounded variation over the admissible policy class: there exists $\Delta_F < \infty$ such that

$$F_k(\pi) - F_k(\pi') \le \Delta_F$$

for all admissible π, π'. Also assume a reference-feasible policy π_{ref} satisfying

$$J_\gamma(\pi_{\text{ref}}) \ge J^\gamma_{\text{ref}} - \delta,$$

so that

$$V^\gamma_\delta(\pi_{\text{ref}}) = 0.$$

By optimality of π_λ,

$$F_k(\pi_\lambda) + \lambda_a V^\gamma_\delta(\pi_\lambda) \le F_k(\pi_{\text{ref}}), \tag{B4}$$

and hence

$$V^\gamma_\delta(\pi_\lambda) \le \frac{\Delta_F}{\lambda_a}. \tag{B5}$$

Defining

$$\xi_\lambda = \frac{\Delta_F}{\lambda_a}, \tag{B6}$$

gives

$$J_\gamma(\pi_\lambda) \ge J^\gamma_{\text{ref}} - \delta - \xi_\lambda. \tag{B7}$$

Equation (B5) bounds reference-relative violation for a global minimizer of the analytical surrogate; it neither guarantees convergence of the nonlinear MARL optimizer nor constraint satisfaction of the implementation-level physical gate.

B.2 Effective Optimization-Level Tolerance

Let

$$J^\star_\gamma = \sup_{\pi \in \Pi} J_\gamma(\pi).$$

From Eq. (B7),

$$J_\gamma(\pi_\lambda) \ge J^\gamma_{\text{ref}} - \delta - \xi_\lambda. \tag{B8}$$

Therefore,

$$J^\star_\gamma - J_\gamma(\pi_\lambda) \le J^\star_\gamma - J^\gamma_{\text{ref}} + \delta + \xi_\lambda. \tag{B9}$$

Define

$$\epsilon^\gamma_{\text{eff}} = J^\star_\gamma - J^\gamma_{\text{ref}} + \delta + \xi_\lambda. \tag{B10}$$

Then

$$J_\gamma(\pi_\lambda) \ge J^\star_\gamma - \epsilon^\gamma_{\text{eff}}. \tag{B11}$$

Thus, the optimization-level tolerance consists of reference suboptimality, the prescribed tolerance, and the finite-penalty residual. This bound characterizes task prioritization in the backbone optimization space and does not by itself establish membership in the evaluation-level set Π_ϵ of Eq. (9).

C Stability of Phase-Adaptive Calibration

C.1 Bounded Phase-Wise Behavioral Calibration

Let the phase-level numerical adapter output be denoted by

$$\xi_k = A_{\mathrm{num}}(x_k),$$

where x_k denotes the phase-level adapter feature vector and ξ_k denotes the resulting calibration state. The main analysis does not require A_{num} to be a trainable neural mapping. In the formal LOCM implementation, the numerical adapter is a trainable MLP updated at phase boundaries. For the stability analysis below, we only require its phase-level output to remain within a bounded calibration range. Let

$$w^{(k)} = g_w(\xi_k) \in \mathcal{W} \subset \mathbb{R}_+^Q,$$

where $\mathcal{W}$ is bounded.

Define

$$\Delta_w = \sup_{u,v\in\mathcal{W}} \|u - v\|_2 < \infty.$$

Then, for any two consecutive phases,

$$\|w^{(k)} - w^{(k-1)}\|_2 \le \Delta_w. \tag{C1}$$

For a fixed trajectory τ, assume that

$$\|\Phi_{\mathrm{obs}}(\tau)\|_2 \le B_\Phi.$$

The phase-dependent behavioral scalarization is

$$C_k(\tau) = w^{(k)\top}\Phi_{\mathrm{obs}}(\tau).$$

Hence,

$$\begin{aligned} |C_k(\tau) - C_{k-1}(\tau)| &= \left|(w^{(k)} - w^{(k-1)})^\top \Phi_{\mathrm{obs}}(\tau)\right| \\ &\le \|w^{(k)} - w^{(k-1)}\|_2 \|\Phi_{\mathrm{obs}}(\tau)\|_2 \\ &\le \Delta_w B_\Phi. \end{aligned} \tag{C2}$$

Thus, the phase-adaptive interface may introduce discrete changes in behavioral calibration, but their magnitude remains bounded on the prescribed calibration range. This is a bounded-adaptation property rather than a smoothness or convergence result.

C.2 Additional Drift of the Penalized Interface Surrogate

We next analyze the phase-to-phase variation of the analytical penalized interface surrogate introduced in Section 3.4. This analysis concerns the scalar penalty representation used for theoretical characterization and does not model the discrete acceptance decisions of the implementation-level physical gate.

For a fixed trajectory τ, define the phase-k LOCM interface contribution as

$$\mathcal{I}_k(\tau) = -\beta w^{(k)\top}\Phi_{\mathrm{obs}}(\tau) - \lambda_a[q_k - G(\tau)]_+, \tag{C3}$$

where

$$q_k = J_{\mathrm{ref},k} - \delta_k$$

denotes the phase-dependent reference threshold. If the reference threshold is fixed, then $q_k = q_{k-1}$ and the corresponding term below vanishes.

Because the positive-part operator $[x]_+ = \max\{x, 0\}$ is 1-Lipschitz,

$$|[a]_+ - [b]_+| \le |a - b|. \tag{C4}$$

Table D1: Within-MASAC near-return matching and behavioral differences.

Statistic	$\leq 1\%$	$\leq 2\%$	$\leq 3\%$	$\leq 5\%$
Raw matched pairs	1324	2631	3272	4542
Deduplicated pairs	1324	2631	3272	4542
Scenario coverage	50/50	50/50	50/50	50/50
Absolute cumulative-return gap	0.051 [0.029, 0.089]	0.123 [0.051, 0.201]	0.164 [0.072, 0.234]	0.222 [0.103, 0.420]
Absolute mean-action-change gap	0.006 [0.005, 0.011]	0.007 [0.004, 0.011]	0.007 [0.003, 0.010]	0.006 [0.003, 0.010]
Absolute counteraction-rate gap	0.015 [0.007, 0.025]	0.014 [0.007, 0.024]	0.014 [0.007, 0.023]	0.014 [0.007, 0.021]
Burden-allocation L_1 gap	0.036 [0.022, 0.046]	0.031 [0.020, 0.043]	0.029 [0.019, 0.042]	0.031 [0.020, 0.043]
Joint executed-action RMSE	0.065 [0.058, 0.073]	0.064 [0.056, 0.071]	0.064 [0.056, 0.071]	0.064 [0.058, 0.070]

For the same fixed trajectory τ,

$$|\mathcal{I}_k(\tau) - \mathcal{I}_{k-1}(\tau)| \leq \beta \left| (w^{(k)} - w^{(k-1)})^\top \Phi_{\text{obs}}(\tau) \right| + \lambda_a |q_k - q_{k-1}|. \tag{C5}$$

Using Eqs. (C1)–(C2),

$$|\mathcal{I}_k(\tau) - \mathcal{I}_{k-1}(\tau)| \leq \beta \Delta_w B_\Phi + \lambda_a |q_k - q_{k-1}|. \tag{C6}$$

Equation (C6) bounds the additional phase-to-phase perturbation introduced by changes in LOCM calibration and, when applicable, the reference-relative outcome-anchor threshold.

Importantly, this bound is evaluated for a fixed trajectory. It does not bound the full objective difference between consecutive training phases because the latter also contains changes induced by the evolving policy, including shifts in the task-return distribution and behavioral trajectory distribution.

Accordingly, Eq. (C6) establishes a bounded-adaptation property for the analytical penalized interface surrogate under the stated assumptions. The bound is evaluated for a fixed trajectory and does not cover policy-induced distribution shift, convergence of the nonlinear MARL optimizer, or the discrete decision logic of the implementation-level physical acceptance gate. It therefore characterizes the stability of the analytical interface representation rather than the complete training dynamics.

D Additional Experimental Results

D.1 Within-MASAC Near-Return Analysis

To examine whether the behavioral heterogeneity observed in RQ1 is solely attributable to differences between MARL algorithms, we repeat the near-return analysis within a single MASAC training run. Distinct late-training MASAC checkpoints are compared under the same 50 evaluation scenarios, using the same return-only matching criterion and behavioral metrics as in Section 4.2. No behavioral quantity is used during matching.

Continuous-valued quantities are reported as median $[Q_1, Q_3]$. Behavioral differences remain nonzero even when comparison is restricted to checkpoints from the same MASAC training run. Although these within-algorithm gaps are smaller than the cross-algorithm differences reported in the main RQ1 analysis, they show that near-return behavioral heterogeneity is not solely induced by architectural differences between MASAC and MAPPO.

D.2 Fixed-Step Open-Loop Intervention

The equal-budget experiment in the main text controls the realized amount of removed action. We additionally report a fixed-step intervention in which actions selected by each diagnostic rule are removed at a prescribed subset of time steps while the original exogenous inputs and open-loop trajectory context are retained.

The evaluated selection criteria are the Φ-based diagnostic, action magnitude, action change, and random selection. This experiment evaluates the local diagnostic value of the behavioral representation rather than the complete training-time LOCM objective.

Table D2: Fixed-step open-loop intervention results.

Algorithm	Selection rule	N	Return non-decrease	Median ΔG	Mean $\Delta G \pm$SD	Δ voltage dev.	Δ loss energy
MASAC	Φ-based diagnostic	23	1.000	0.291 [0.273, 0.297]	0.287 ± 0.015	-0.498 [-0.671, -0.494]	-0.522 [-0.592, -0.520]
MASAC	Action magnitude	23	1.000	0.299 [0.265, 0.303]	0.295 ± 0.054	-0.190 [-0.245, -0.188]	-0.739 [-0.744, -0.717]
MASAC	Action change	23	1.000	0.038 [0.034, 0.040]	0.041 ± 0.010	-0.425 [-0.427, -0.158]	-0.595 [-0.607, -0.590]
MASAC	Random	69	0.594	0.087 [-0.080, 0.247]	0.065 ± 0.260	-0.024 [-0.206, 0.233]	-0.306 [-0.398, -0.191]
MAPPO	Φ-based diagnostic	12	0.000	-0.001 [-0.001, -0.001]	-0.001 ± 0.000	0.001 [0.001, 0.001]	-0.000 [-0.000, -0.000]
MAPPO	Action magnitude	12	1.000	0.001 [0.001, 0.001]	0.001 ± 0.000	-0.002 [-0.002, -0.002]	-0.003 [-0.003, -0.003]
MAPPO	Action change	12	0.000	-0.001 [-0.001, -0.001]	-0.001 ± 0.000	0.001 [0.001, 0.001]	-0.001 [-0.001, -0.001]
MAPPO	Random	36	0.278	-0.001 [-0.001, 0.000]	-0.001 ± 0.001	0.001 [-0.000, 0.002]	-0.001 [-0.002, -0.001]

Table D3: Trajectory-level paired intervention results.

Algorithm	Paired comparison	N	Wins	Ties	Losses	Median paired ΔG
MASAC	Φ-based vs. Action magnitude	23	21	0	2	0.035423
MASAC	Φ-based vs. Action change	23	23	0	0	0.125250
MASAC	Φ-based vs. Random	23	20	0	3	0.137380
MAPPO	Φ-based vs. Action magnitude	12	0	0	12	-0.001176
MAPPO	Φ-based vs. Action change	12	12	0	0	0.000339
MAPPO	Φ-based vs. Random	12	6	0	6	0.000113

For the random criterion, N includes three random intervention repeats per base trajectory. Under MASAC, all three structured selection rules achieve a 100% return non-decrease rate, but produce different trade-offs among return, voltage deviation, network loss, and SoC-related effects. The MAPPO interventions remain numerically small, consistent with the substantially lower action activity of the matched MAPPO trajectories.

D.3 Trajectory-Level Paired Intervention Analysis

Aggregate statistics may obscure whether an intervention criterion performs consistently on the same trajectories. We therefore conduct paired comparisons in which two selection rules are evaluated on identical saved trajectories.

For each pair, a win indicates that the Φ-based diagnostic produces a larger return change than the corresponding alternative criterion.

For MASAC, the Φ-based diagnostic outperforms action magnitude on 21 of 23 trajectories, action change on all 23 trajectories, and random selection on 20 of 23 trajectories. The MAPPO results are more mixed and remain close to zero in absolute magnitude, again reflecting the low control activity in the corresponding trajectories. These results support the use of the behavioral representation as a local diagnostic, without implying universal dominance over alternative intervention rules.

D.4 Detailed Results under the Standard MEDIUM Protocol

Table D4 reports the full set of metrics corresponding to the Standard MEDIUM evaluation summarized in Section 4.4.1 and Table 3.

Across all three backbones, LOCM reduces action magnitude and temporal variation. MASAC-LOCM also improves environmental return and substantially reduces line loss, whereas HAPPO-LOCM and MADPO-LOCM exhibit modest return decreases. These results support a backbone-dependent task–behavior trade-off rather than uniform improvement of every metric.

D.5 Independent Training-Seed Evaluation

We further evaluate whether the behavioral-refinement effect persists across independent training realizations. HAPPO, HAPPO-LOCM, MADPO, and MADPO-LOCM are each trained with three independent seeds (3, 4, and 5). Each independently trained model is treated as one experimental unit: evaluation metrics are first aggregated over the corresponding evaluation scenarios within each trained model, after which the cross-seed mean and standard deviation are computed.

The results below correspond to the same independent-seed evaluation reported in Section 4.4.2.

Table D4: Performance under the Standard MEDIUM protocol.

Method	Return G	Line loss	Action mag.	Action variation	Counteraction rate	Role-switch rate
MASAC	-12.059	1.798	0.192	0.211	0.656	0.537
MASAC-LOCM	-11.559	1.252	0.048	0.040	0.719	0.463
HAPPO	-11.132	2.013	0.307	0.332	0.646	0.505
HAPPO-LOCM	-11.185	2.002	0.258	0.268	0.521	0.421
MADPO	-10.784	1.982	0.301	0.338	0.635	0.484
MADPO-LOCM	-11.145	1.935	0.246	0.259	0.490	0.474

Table D5a: Independent training-seed task-level evaluation (mean $\pm$ SD, $n = 3$).

Method	Return	Voltage dev.	Cumulative loss	Max voltage	Min voltage
HAPPO	-12.6107 ± 0.0812	0.2124 ± 0.0052	4.4964 ± 0.3754	1.0294 ± 0.0051	0.9260 ± 0.0118
HAPPO-LOCM	-12.4348 ± 0.4225	0.2122 ± 0.0096	4.3160 ± 0.3080	1.0267 ± 0.0070	0.9343 ± 0.0075
MADPO	-12.5793 ± 0.3200	0.2143 ± 0.0050	4.4999 ± 0.4309	1.0294 ± 0.0057	0.9240 ± 0.0141
MADPO-LOCM	-12.5586 ± 0.5271	0.2140 ± 0.0111	4.3673 ± 0.3484	1.0299 ± 0.0071	0.9354 ± 0.0077

D.5.1 Task-Level Metrics

D.5.2 Behavioral Metrics

Relative to the corresponding baseline, mean action variation decreases by approximately 20.8% for HAPPO and 12.8% for MADPO. Action magnitude is also reduced for both backbones. At the same time, mean environmental return and voltage deviation remain comparable. These results support cross-seed reproducibility of the behavioral-refinement effect rather than improved training stability; in particular, the return standard deviation is not uniformly reduced by LOCM.

D.6 Ablation Analysis

We report on the complete component-level metrics corresponding to the compact ablation in the main paper. Rule-diagnoser replaces the LLM-assisted semantic diagnosis with deterministic rule-based diagnosis; No-coord removes the behavioral-refinement component; Static-coord uses fixed behavioral calibration weights; No-adapter removes the trainable numerical adapter; and No-anchor removes the reference-relative task-preservation mechanism. LOCM denotes the full model.

Task-Level Metrics

Behavioral Metrics

These results show complementary rather than uniformly monotonic component effects. Removing behavioral refinement altogether produces the largest degradation in action magnitude and temporal variation, while static weighting, rule-based diagnosis, removal of the numerical adapter, and removal of the outcome anchor affect different subsets of task and behavioral metrics.

D.7 Full Traffic-Control Evaluation

We report the complete traffic-control results under unseen traffic sequences. For terminological consistency across the evaluated domains, the traffic-control metric labeled Smoothness is reported here as Action variation.

Across all three traffic-demand levels, LOCM reduces average delay, queue overflow, maximum queue occupancy, green variance, action variation, action magnitude, and counteraction while increasing throughput. In contrast, role switching and burden trend increase under LOCM. This pattern indicates that the behavioral metrics capture distinct execution characteristics rather than collapsing into a single monotonic notion of coordination quality.

D.8 Comparison of Calibration Strategies

To examine the contribution of phase-level calibration, we compare the full LLM-assisted LOCM with three alternative calibration strategies: fixed calibration (LOCM-Fixed), deterministic rule-

Table D5b: Independent training-seed behavioral evaluation (mean ± SD, $n = 3$).

Method	Action mag.	Action variation
HAPPO	0.1505 ± 0.0552	0.0437 ± 0.0276
HAPPO-LOCM	0.1309 ± 0.0117	0.0346 ± 0.0095
MADPO	0.1430 ± 0.0493	0.0415 ± 0.0255
MADPO-LOCM	0.1289 ± 0.0025	0.0362 ± 0.0073

Table D6a: Full task-level component ablation.

Variant	Max voltage	Min voltage	Voltage dev.	Network loss	SoC var.
Rule-diagnoser	1.1019 ± 0.0074	0.9239 ± 0.0005	0.2280 ± 0.0119	0.0726 ± 0.0025	0.0333 ± 0.0024
No-coord	1.1825 ± 0.0111	0.8436 ± 0.0256	0.2594 ± 0.0076	0.1680 ± 0.0047	0.0190 ± 0.0046
Static-coord	1.0930 ± 0.0070	0.9029 ± 0.0129	0.2201 ± 0.0092	0.0735 ± 0.0010	0.0412 ± 0.0023
No-adapter	1.0901 ± 0.0104	0.9361 ± 0.0074	0.2122 ± 0.0058	0.0741 ± 0.0009	0.0345 ± 0.0017
No-anchor	1.0891 ± 0.0100	0.9266 ± 0.0164	0.2147 ± 0.0070	0.0736 ± 0.0017	0.0328 ± 0.0015
LOCM	1.0735 ± 0.0032	0.9432 ± 0.0013	0.1966 ± 0.0019	0.0537 ± 0.0003	0.0119 ± 0.0007

based calibration (LOCM-Rule), and shuffled semantic calibration (LOCM-Shuffled). LOCM-Shuffled preserves LLM-assisted semantic diagnosis but deliberately breaks the temporal alignment between the diagnosis generated for the current phase and the calibration applied to the subsequent phase. At each phase boundary, a semantic diagnosis is generated for the current phase. Before this diagnosis is added to the diagnosis pool, one diagnosis is sampled uniformly at random from the pool containing diagnoses from previous phases, and the sampled historical diagnosis is mapped to the behavioral calibration parameters used in the subsequent phase. The current diagnosis is appended to the pool only after the sampling step and therefore cannot be selected immediately. When the historical diagnosis pool is empty in the first phase, a predefined medium-level diagnosis is used. The sampling sequence is reproducible for each training seed using a dedicated random-number generator initialized with the training seed plus a fixed offset of 910003. The underlying HAPPO backbone is included as a reference.

All LOCM variants use the same observable behavioral representation and downstream behavioral-refinement framework and differ primarily in how phase-level calibration is determined. The purpose of this comparison is to characterize how different calibration mechanisms affect the task–behavior trade-off, rather than to establish LLM assistance as a necessary condition for behavioral refinement. All methods are evaluated under the same standard power-distribution evaluation protocol, and values are reported as mean ± SD over 10 fixed evaluation seeds.

All four LOCM variants improve the reported mean task, physical, and execution-level metrics relative to the underlying HAPPO backbone, indicating that the trajectory-level behavioral-refinement mechanism does not depend exclusively on the LLM interface. At the same time, no single calibration strategy uniformly dominates across all criteria. LOCM-Fixed attains the highest mean environmental return and the largest reductions in action magnitude, temporal action variation, counteraction, and SoC penalty. LOCM-Shuffled achieves the lowest maximum voltage and dominant-agent switching rate, while LOCM-Rule provides another competitive operating point.

The full LLM-assisted configuration exhibits a different trade-off. LOCM-Full achieves the lowest voltage deviation and network loss and ties for the highest minimum voltage, while retaining substantial reductions in action magnitude and temporal variation relative to HAPPO. Compared with LOCM-Shuffled, the phase-aligned LLM-assisted configuration further reduces voltage deviation and network loss, although the shuffled variant remains competitive or better on several individual execution metrics. This indicates that phase-specific semantic alignment primarily affects the operating point of the task–behavior trade-off rather than producing uniform gains across all behavioral coordinates.

Overall, these results provide supporting evidence for LLM-assisted phase-adaptive calibration while also showing that the underlying behavioral-refinement mechanism remains effective under fixed, rule-based, or semantically shuffled calibration. The role of the LLM should therefore be interpreted as providing an adaptive semantic interface for selecting among behavioral preferences,

Table D6b: Full behavioral component ablation.

Variant	Counteraction	Action mag.	Burden trend	Action variation	Role switch
Rule-diagnoser	0.7188 ± 0.0312	0.0869 ± 0.0012	-0.3985 ± 0.0569	0.0948 ± 0.0024	0.6456 ± 0.0425
No-coord	0.5590 ± 0.0335	0.2456 ± 0.0043	-0.1920 ± 0.0223	0.3284 ± 0.0110	0.5930 ± 0.0061
Static-coord	0.7535 ± 0.0241	0.0928 ± 0.0028	-0.1861 ± 0.0718	0.0957 ± 0.0025	0.6842 ± 0.0365
No-adapter	0.7847 ± 0.0318	0.0871 ± 0.0040	-0.1791 ± 0.0481	0.0912 ± 0.0019	0.6175 ± 0.0219
No-anchor	0.7604 ± 0.0521	0.0889 ± 0.0013	-0.1536 ± 0.0786	0.0933 ± 0.0018	0.6526 ± 0.0421
LOCM	0.6736 ± 0.0241	0.0471 ± 0.0003	-0.2140 ± 0.0146	0.0416 ± 0.0029	0.5123 ± 0.0530

Table D7: Traffic signal control under unseen traffic sequences (mean $\pm$ SD).

Metric	Low SAC	Low LOCM	Medium SAC	Medium LOCM	High SAC	High LOCM
Avg. delay	0.4569 ± 0.0096	0.3735 ± 0.0054	0.5697 ± 0.0029	0.4859 ± 0.0033	0.6891 ± 0.0138	0.5911 ± 0.0036
Queue overflow	2.4317 ± 0.3861	2.3211 ± 0.2499	4.3466 ± 0.3421	4.0683 ± 0.3080	6.1783 ± 0.3493	5.7548 ± 0.4106
Throughput	72.5190 ± 0.2750	72.9210 ± 0.4304	69.4050 ± 0.2762	69.8490 ± 0.3363	66.1490 ± 0.1952	66.5800 ± 0.2438
Max. queue	0.6001 ± 0.0072	0.5718 ± 0.0064	0.6537 ± 0.0029	0.6407 ± 0.0025	0.6963 ± 0.0053	0.6856 ± 0.0070
Green variance	0.0086 ± 0.0006	0.0073 ± 0.0007	0.0088 ± 0.0004	0.0082 ± 0.0004	0.0099 ± 0.0012	0.0095 ± 0.0013
Action variation	0.0909 ± 0.0062	0.0264 ± 0.0023	0.0911 ± 0.0055	0.0318 ± 0.0031	0.0922 ± 0.0028	0.0361 ± 0.0049
Action magnitude	0.0593 ± 0.0025	0.0173 ± 0.0007	0.0598 ± 0.0032	0.0208 ± 0.0014	0.0607 ± 0.0024	0.0247 ± 0.0025
Counteraction	0.5764 ± 0.0217	0.0694 ± 0.0159	0.5486 ± 0.0335	0.0903 ± 0.0434	0.5903 ± 0.0535	0.1389 ± 0.0524
Role switch	0.3158 ± 0.0279	0.4421 ± 0.0586	0.3193 ± 0.0304	0.4316 ± 0.0526	0.2982 ± 0.0370	0.4211 ± 0.0211
Burden trend	0.1418 ± 0.0290	0.4977 ± 0.6058	0.1235 ± 0.0579	0.4360 ± 0.1257	0.0934 ± 0.0504	0.5239 ± 0.1906

Table D8a: Task and physical performance under different calibration strategies (mean $\pm$ SD over 10 fixed evaluation seeds).

Method	Return G	Voltage dev.	Network loss (MWh)	Max. voltage	Min. voltage
HAPPO	-11.1562 ± 0.5217	18.0690 ± 0.1867	2.0785 ± 0.0433	1.0908 ± 0.0067	0.8996 ± 0.0084
LOCM-Fixed	-10.5839 ± 0.5073	17.7139 ± 0.2385	1.9501 ± 0.0339	1.0824 ± 0.0062	0.9282 ± 0.0071
LOCM-Rule	-10.7833 ± 0.5223	17.6890 ± 0.2433	1.9776 ± 0.0330	1.0810 ± 0.0058	0.9292 ± 0.0071
LOCM-Shuffled	-10.7777 ± 0.5106	17.7104 ± 0.2337	1.9658 ± 0.0351	1.0792 ± 0.0059	0.9288 ± 0.0071
LOCM-Full	-10.6633 ± 0.5055	17.5814 ± 0.2173	1.9410 ± 0.0349	1.0798 ± 0.0055	0.9292 ± 0.0072

Table D8b: Execution-level and auxiliary metrics under different calibration strategies (mean $\pm$ SD over 10 fixed evaluation seeds).

Method	Action mag.	Action variation	Counteraction	Dominant-agent switching	SoC penalty
HAPPO	0.3154 ± 0.0074	0.3519 ± 0.0109	0.6562 ± 0.0264	0.5221 ± 0.0416	1.4518 ± 0.2685
LOCM-Fixed	0.2597 ± 0.0056	0.2744 ± 0.0098	0.5677 ± 0.0394	0.4400 ± 0.0548	0.5535 ± 0.2495
LOCM-Rule	0.2676 ± 0.0053	0.2808 ± 0.0112	0.6208 ± 0.0237	0.4368 ± 0.0447	1.0893 ± 0.2738
LOCM-Shuffled	0.2732 ± 0.0053	0.2862 ± 0.0106	0.6240 ± 0.0211	0.4116 ± 0.0609	1.0432 ± 0.2979
LOCM-Full	0.2632 ± 0.0066	0.2859 ± 0.0121	0.6323 ± 0.0260	0.4568 ± 0.0474	0.9507 ± 0.2939

rather than as a necessary condition for trajectory-level behavioral refinement or as a mechanism that uniformly improves every reported metric.

E Experimental and Implementation Details

This appendix summarizes the environment configurations, backbone hyperparameters, LOCM-specific settings, language-model interface, and evaluation protocols used in the reported experiments. The power-distribution settings correspond to the implementation used for the main power-control experiments. For the cross-domain traffic-control evaluation, we report the configuration used to generate the results in Section 4.4.3 and Appendix D.7.

E.1 Power Distribution Network Environment

The power-distribution task is formulated as a continuous-action cooperative multi-agent control problem on a modified IEEE 33-bus distribution system. Three controllable storage agents are deployed at selected buses and regulate their charging/discharging actions according to local temporal, renewable-generation, and state-of-charge information.

Each episode contains 96 control intervals with a 15-minute resolution. The environment uses AC power flow through pandapower. The executed actions may differ from the nominal policy outputs under the perturbed protocol because of execution delay, actuation noise, and packet dropout.

Table E1: Power-distribution environment settings.

Item	Setting
Benchmark system	Modified IEEE 33-bus distribution network
Power-flow solver	pandapower AC power flow
Number of agents	3
Controlled buses	17, 19, 21
Controllable device	One storage agent at each controlled bus
PV buses	9, 13, 23, 24, 22, 32, 31, 12
Wind-generation buses	16, 17, 19, 20, 21, 26, 27
Action space	Continuous
Normalized policy action	$[-1, 1]$
Physical action scale	2.0
Internal SoC capacity parameter	7.7
Terminal SoC threshold	0.5
Episode length	96 steps
Time resolution	0.25 h
Actor observation	Normalized time index, local SoC, load, PV generation, and wind generation
Critic augmentation	Aggregate action, counteraction statistic, mean SoC, and SoC standard deviation
Evaluation voltage criterion	$V_{\max} > 1.05$ p.u. or $V_{\min} < 0.95$ p.u.
Recorded physical metrics	Voltage deviation, network loss, maximum voltage, minimum voltage, and SoC-related metrics

Table E2: Standard MEDIUM perturbation settings.

Item	Setting
Observation-noise standard deviation	0.02
Execution delay	1 step
Execution-noise standard deviation	0.06
Execution-dropout probability	0.10
Reward noise	Student-t, $df = 3$, scale $= 0.04$
Episode-drift intensity	0.018
Curriculum	Enabled
Maximum difficulty amplitude	0.9
Voltage-deviation reward weight	0.6
SoC reward weight	0.4
Terminal SoC penalty scale	40.0

The original environmental reward prioritizes voltage regulation and terminal SoC feasibility,

$$r_t^{\text{env}} = -\left(w_v \ell_{v,t} + w_s \ell_{\text{soc},t}\right),$$

with $w_v = 0.6$ and $w_s = 0.4$ under the standard configuration. The terminal SoC penalty uses a scale factor of 40.0.

The standard MEDIUM protocol additionally introduces observation, communication, execution, and reward perturbations. The corresponding settings are summarized below.

The monthly evaluation uses 31 daily operating sequences, each containing 96 control intervals. All evaluated policies are executed deterministically on the same daily exogenous sequences. The daily sequences are evaluation scenarios and are therefore not treated as independent training realizations. The power-network configuration and MEDIUM perturbation settings reported above are those used in the corresponding power-distribution experiments.

E.2 Traffic-Control Environment

The traffic-control task provides a discrete-action cross-domain evaluation. The environment contains three controlled intersections, each represented by one agent. Each agent selects a discrete modification to its current green-split ratio. Before behavioral statistics are computed, the categorical action is decoded into the corresponding physical green-split increment.

The environment and SAC configuration reported below are those used for the cross-domain traffic-control evaluation in Section 4.4.3 and Appendix D.7.

Table E3: Traffic-control environment settings.

Item	Setting
Number of intersections / agents	3
Controlled intersections	0, 1, 2
Action space	Discrete
Actions per agent	5
Physical action set	Keep, $+0.05$, -0.05, $+0.10$, -0.10 green-split change
Initial green split	0.50
Minimum green split	0.15
Maximum green split	0.85
Maximum queue capacity	80.0
Saturation rate	28.0
Intersection-coupling rate	0.6
Base arrival coefficient	15.0
Episode length	96 steps
Recorded task metrics	Average delay, queue overflow, maximum queue ratio, throughput, and green-split variance

Table E4: Traffic-control perturbation and reward settings.

Item	Setting
Observation-noise standard deviation	0.02
Execution delay	0 steps
Execution-noise standard deviation	0.06
Execution-dropout probability	0.10
Reward noise	Student-t, $df = 3$, scale $= 0.04$
Episode-drift intensity	0.018
Curriculum	Enabled
Maximum difficulty amplitude	0.9
Delay weight in baseline reward	0.6
Green-split penalty weight	0.4
Green-split penalty scale	30.0
Queue-overflow penalty	$5\times$ overflow amount
Queue-warning threshold	$Q_{\max} > 0.85$
Absolute safety criterion	$Q_{\max} > 0.90$ or delay > 0.70
Relative outcome reference	Delay and queue overflow relative to the baseline reference

The categorical output is used only for policy decision making. For the behavioral representation, each categorical choice is converted to its corresponding signed green-split increment,

$$a_t^{(i)} \in \{0, +0.05, -0.05, +0.10, -0.10\},$$

and the resulting physical increment sequence is used to compute action variation, action magnitude, counteraction, role switching, and the remaining execution-level statistics.

The traffic experiments evaluate low-, medium-, and high-demand sequences. The same evaluation protocol is applied to SAC and LOCM, and the reported statistics are computed under matched traffic conditions.

E.3 MARL Backbone Hyperparameters

The power-distribution experiments use MASAC, MAPPO, HAPPO, and MADPO as the underlying MARL backbones. The traffic-control experiment uses a discrete-action SAC backbone. Backbone-specific optimization is performed through the native training objective of each algorithm. LOCM modifies the secondary behavioral preference and outcome-anchoring terms without replacing the underlying optimizer.

The discount factor $\gamma = 0.99$ is used only by the native MARL backbones for policy optimization. In contrast, the undiscounted episodic environmental return is used for return-equivalence analysis and task-level evaluation throughout this paper.

MAPPO collects 10 episodes per rollout update and performs five PPO epochs using minibatches of 128 samples. HAPPO and MADPO collect four episodes per rollout update and perform five opti-

Table E5: Backbone-specific training hyperparameters.

Parameter	MASAC	MAPPO	HAPPO	MADPO	Traffic SAC
Actor learning rate	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	3×10^{-4}
Critic learning rate	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	3×10^{-4}
Temperature learning rate	3×10^{-4}	N/A	N/A	N/A	3×10^{-4}
Optimizer	Adam	Adam	Adam	Adam	Adam
Discount factor γ	0.99	0.99	0.99	0.99	0.99
GAE λ	N/A	0.95	0.95	0.95	N/A
PPO clip ratio	N/A	0.2	0.2	0.2	N/A
Entropy coefficient	Automatic	0.01 with decay	0.01	0.01	Automatic
Initial SAC temperature α	0.2	N/A	N/A	N/A	0.2
Batch / minibatch size	256	128	Full rollout	Full rollout	256
Rollout episodes per update	N/A	10	4	4	N/A
Optimization epochs per rollout	N/A	5	5	5	N/A
Replay-buffer size	500,000	N/A	N/A	N/A	500,000
Soft-update coefficient τ	0.005	N/A	N/A	N/A	0.005
SAC updates per environment step	1	N/A	N/A	N/A	1
Warm-up steps	5,000	N/A	N/A	N/A	5,000
Hidden layers	3	2	2	2	3
Hidden dimension	256	128	256	256	256
Activation	ReLU	ReLU	ReLU	ReLU	ReLU
Continuous actor	Tanh-squashed Gaussian	N/A	N/A	N/A	N/A
Discrete actor	N/A	N/A	N/A	N/A	Categorical
Gradient clipping	Enabled	Enabled	Enabled	Enabled	Enabled
Maximum gradient norm	1.0	10.0	10.0	10.0	1.0

mization epochs over the full collected rollout, with no additional minibatch subdivision. Since each power-distribution episode contains 96 environment steps, a HAPPO or MADPO rollout typically contains 384 joint environment time steps.

For the power-control MASAC implementation, automatic entropy tuning uses a target entropy of $-a_{\mathrm{dim}}$ per agent, corresponding to -1.0 for the one-dimensional agent action used in the reported experiments.

For the SAC implementations, double Q-networks, target networks, automatic entropy tuning, and soft target updates are used. In the traffic-control setting, the actor outputs a categorical distribution over the five discrete actions and the critic estimates Q-values for the complete action set.

E.4 LOCM Phase-Adaptive Calibration

LOCM introduces a secondary trajectory-level behavioral representation on top of the original task objective. For the evaluated control tasks, the observable representation contains three behavioral coordinates corresponding to action-sign counteraction, dominant-agent switching, and temporal action variation.

For consistency with earlier experimental logging, the terms "role switch" and "role-switch rate" used in some result tables refer to the same dominant-agent switching statistic defined in Eq. (8), rather than to a separately learned latent role.

Behavioral statistics are aggregated over each training phase before semantic and numerical calibration. The LLM operates only on phase-level summaries and produces a structured semantic diagnosis, which is converted into a numerical supervision target. A trainable MLP numerical adapter maps the phase-level feature vector to continuous control variables, with the numerical target providing supervision at phase boundaries. The LLM does not participate in real-time action generation.

The semantic interface first produces a structured diagnosis, which is converted into a numerical supervision target. The trainable MLP adapter then produces continuous phase-level control variables. In the implemented MASAC-LOCM path, its first three components are bounded behavioral penalty coefficients associated with action-sign counteraction, dominant-agent switching, and temporal action variation, and are used directly without simplex normalization. The remaining two components correspond to action intensity and collective pull and are used only by the implementation-level action-style pathway. These two auxiliary variables are excluded from the trajectory-level behavioral representation Φ_{obs}, the geometric refinement result in Theorem 1, and the behavioral diagnostics used in RQ1–RQ2.

Table E6: LOCM implementation and calibration settings.

Item	Setting
Behavioral coordinates	Counteraction, dominant-agent switching, temporal action variation
Phase length	100 episodes
Phase update frequency	Once every 100 episodes
Phase summary content	Task performance, behavioral statistics, phase trends, and previous calibration state
Numerical-adapter type	Trainable MLP
Numerical-adapter input dimension	15 dims: 8 phase statistics + 2 reference ratios + 5 previous controls
Numerical-adapter hidden dimension	64
Numerical-adapter hidden layers	2
Adapter optimizer	Adam
Adapter learning rate	1×10^{-3}
Adapter optimizer steps per phase boundary	1
Adapter loss	MSE between adapter output and numerical supervision target
Adapter gradient clipping	Enabled
Maximum adapter gradient norm	1
Adapter output	3 behavioral penalty coefficients + 2 auxiliary action-style variables
Reference source	Baseline-policy performance
Reference statistics	Average voltage deviation and average line loss
Behavioral update timescale	Phase level
Policy update timescale	Environment / backbone update timescale
LLM intervention in real-time control	None

Table E7: LLM semantic-interface settings.

Item	Setting
Model	deepseek-V4-PRO
Interface	OpenAI-compatible chat-completion API
Invocation frequency	Once per phase
Phase length	100 episodes
Temperature	0.4
Maximum output tokens	2500
Top-p	Default API setting
Output format	Structured JSON
Failure handling	Rule-based semantic fallback
Real-time action generation	No
Direct numerical reward generation	No
Role in LOCM	Phase-level semantic diagnosis

The adapter output remains fixed within each phase and is updated only at phase boundaries. The adapter is therefore not a real-time controller, and language-model inference remains outside the environment control loop.

The reference-relative outcome anchor prioritizes preservation of task performance with respect to an attainable baseline reference. Sections 3.3–3.4 and Appendix B represent this principle through an analytical penalized surrogate, whereas the reported power-distribution implementation operationalizes it through a reference-relative physical acceptance gate over task-relevant quantities. The gate determines whether the current behavioral regime remains admissible for continued refinement; it is not mathematically identical to the analytical penalty surrogate and should not be interpreted as an exact constrained optimizer. Further implementation details are provided in Appendix F.1 and Section F.3.2.

E.5 LLM Semantic Interface

The LLM is used exclusively as a phase-level semantic diagnostic interface. It receives aggregated task and behavioral statistics from the completed training phase and produces a structured semantic diagnosis of the current behavioral regime. The LLM neither generates real-time actions nor directly specifies the numerical coefficients used by the MARL optimizer.

The semantic interface is implemented through a chat-completion API with a system message and a structured phase-level user message. LLM inference is invoked only at phase boundaries and remains outside the environment-step control loop.

Table E8: Training and evaluation protocols.

Item	Setting
RQ1 late-training checkpoints	15 MASAC and 15 MAPPO checkpoints
RQ1 evaluation scenarios	50 matched scenarios
RQ1 matching variable	Undiscounted cumulative environmental return only
Behavioral variables used for matching	None
RQ1 tolerance levels	1%, 2%, 3%, and 5%
Within-MASAC check	Distinct late-training MASAC checkpoints under the same 50 scenarios
RQ2 intervention	Open-loop intervention on saved near-return trajectories
Equal-budget removal	10% of cumulative action L_1 magnitude, $B = 0.10 \sum_t \|u_t\|_1$
Supplementary intervention	Fixed-step action removal
Standard power evaluation	MEDIUM perturbation protocol over 10 fixed evaluation seeds
Monthly power evaluation	31 daily sequences $\times$ 96 steps
Monthly statistical unit	One operating day / scenario
Independent-training-seed analysis	HAPPO, HAPPO-LOCM, MADPO, and MADPO-LOCM
Independent training seeds	3 per method (seeds 3, 4, and 5)
Cross-seed aggregation	First aggregate evaluation scenarios within each training seed, then compute mean $\pm$ SD across training seeds
Traffic-control evaluation	Low-, medium-, and high-demand unseen traffic sequences
Traffic baseline	Discrete-action SAC
Behavioral analysis for traffic	Decoded physical green-split increments
Evaluation policy	Deterministic unless otherwise specified

The LLM input summarizes both task and behavioral evidence from the current phase. The prompt package includes:

- phase-level task and behavioral statistics;
- representative trajectories across distinct task and behavioral regimes;
- boundary or transition trajectories;
- recurring execution patterns requiring attention;
- emerging execution patterns of potential interest;
- comparison with the preceding phase;
- accumulated historical phase information; and
- similar previously observed phases when available.

The structured semantic output describes the current behavioral regime, dominant execution patterns, qualitative behavioral preferences, uncertainty, and the recommended semantic emphasis for the subsequent phase. This semantic portrait is not used directly as a numerical learning signal. Instead, it is converted into a numerical supervision target for the phase-boundary adapter update, while the adapter itself operates on the phase-level feature vector. The behavioral-calibration component of these variables determines the phase-dependent behavioral weights used by the downstream learning process.

If the LLM call fails or the returned object cannot be parsed, a deterministic rule-based fallback produces a compatible structured semantic diagnosis from the same phase-level summary. The fallback output is passed through the same downstream numerical-adaptation pipeline, allowing training to continue without an online LLM response. Thus, both the LLM-assisted and fallback paths remain confined to phase-level semantic diagnosis and do not intervene directly in environment-step action selection.

E.6 Training and Evaluation Protocol

The evaluation is organized according to the three research questions in the main paper. Unless otherwise stated, policies are evaluated deterministically and methods being compared within the same experiment are exposed to matched initial conditions and exogenous inputs.

For RQ1, near-return matching is performed exclusively from cumulative environmental return. The behavioral quantities are computed only after matching and therefore cannot influence pair selection. The symmetric relative return gap is $d_G(a, b) = |a - b| / \max((|a| + |b|)/2, 10^{-8})$.

Raw matched pairs are deduplicated according to trajectory-output equivalence rather than file-row identity. Within each algorithm and scenario, two trajectory records are assigned to the same output group when their arrays for executed actions, proposed actions, reward, voltage deviation, state of charge, SoC penalty, network loss, and recorded features have identical shapes

and satisfy `numpy.allclose` with absolute tolerance 10^{-12}, relative tolerance 10^{-10}, and `equal_nan=True`. A matched pair is then retained only once for each unique tuple consisting of the scenario identifier and the two trajectory-output group identifiers.

For RQ2, the saved trajectories are subjected to matched open-loop interventions. The environment remains unchanged, allowing the intervention to isolate the local effect of removing selected execution activity without introducing closed-loop policy adaptation. For the equal-budget intervention, the removal budget for each original trajectory is $B = 0.10 \sum_t \|u_t\|_1$. Time steps are ranked by the diagnostic-specific selection score using a stable descending sort. A selected step is set to zero when its complete action magnitude fits within the remaining budget. If the remaining budget is smaller than the next selected step's magnitude, that step is uniformly scaled across agents so that the removed L_1 action magnitude exactly matches the remaining budget. Consequently, the intervention removes 10% of the original trajectory's cumulative absolute action magnitude up to numerical tolerance, rather than simply removing 10% of time steps.

For the standard power-distribution evaluation, all compared methods are evaluated deterministically over the same 10 fixed evaluation seeds.

For the monthly evaluation, the 31 days represent repeated operating scenarios rather than 31 independent training runs. Consequently, day-level variation is interpreted as variation across operating conditions, not as training-seed uncertainty.

For independent-seed experiments, the unit of statistical analysis is the independently trained model. Metrics are first averaged over the corresponding evaluation scenarios for each trained seed, and cross-seed mean and standard deviation are then computed. This avoids treating evaluation scenarios from the same trained policy as independent training samples.

For traffic control, the same unseen low-, medium-, and high-demand sequences are used for SAC and LOCM, and all behavioral metrics are evaluated from the decoded physical actions rather than the categorical policy labels.

F Additional Reproducibility Materials

F.1 Phase-Adaptive LOCM Training Procedure

LOCM operates on a slower calibration timescale than the underlying MARL backbone. The task controller itself remains the original MARL policy, while LOCM periodically summarizes executed trajectories, diagnoses the current behavioral regime, and updates the phase-level behavioral calibration.

Training first establishes a task-performance reference using the original environmental objective. During behavioral refinement, trajectories are collected over a phase of completed episodes. For each trajectory, LOCM evaluates the environmental outcome together with the observable behavioral representation, whose three coordinates correspond to action-sign counteraction, dominant-agent switching, and temporal action variation.

The resulting trajectory statistics are aggregated into a phase summary s_k. The LLM then produces a structured semantic diagnosis p_k, which is converted into a numerical supervision target y_k. If the LLM is unavailable or its output cannot be parsed, a deterministic rule-based fallback produces the same structured semantic interface before numerical adaptation.

The trainable MLP adapter processes the phase-level feature vector and produces the control variables for the subsequent phase, while the numerical target supervises the adapter update.

The first three components of c_k determine the behavioral weights, while the remaining two are auxiliary action-style variables for the practical implementation. The adapter is updated only at phase boundaries, and the LLM neither generates real-time actions nor directly specifies the numerical reward coefficients used by the MARL backbone.

Algorithm F1. Phase-Adaptive Training with LOCM

Input:

MARL backbone π_θ

environment E

phase length K

observable behavioral representation Φ_{obs}

reference task-performance statistics J_{ref}

LLM semantic interface P_{LLM}

trainable numerical adapter A_ϕ

Initialize:

backbone parameters θ

adapter parameters ϕ

calibration state c_0

reference task-performance statistics J_{ref}

1. Establish the task-performance reference using the original environmental objective.
2. for phase $k = 1, 2, \ldots$ do
3. Execute the current MARL policy and collect K completed episodes.
4. For each trajectory τ, compute the environmental outcome and the observable behavioral coordinates.
5. Aggregate trajectory-level task and behavioral statistics:
$$s_k = \mathcal{S}(\mathcal{B}_k).$$
6. Construct the phase-level semantic prompt from task-performance statistics, behavioral statistics, representative trajectories, previous-phase comparison, and historical phase information.
7. Query the semantic interface:
$$p_k = P_{\text{LLM}}(s_k).$$
8. Convert the structured semantic diagnosis into a numerical supervision target:
$$y_k = h(p_k).$$
9. If the LLM call or parsing fails, replace the LLM output with a deterministic rule-based semantic diagnosis,
$$p_k \leftarrow \text{RuleBasedDiagnosis}(s_k),$$
and compute
$$y_k = h(p_k).$$
10. Form x_k, update the numerical adapter toward y_k, and obtain
$$c_k = A_\phi(x_k).$$
11. Extract the first three components as behavioral calibration coefficients, and route the remaining two components to the implementation-level action-style pathway controlling action intensity and collective pull.
$$w^{(k)} = g_w(c_k^{\text{beh}}).$$
12. Construct the phase-dependent behavioral refinement signal C_k, using the backbone-specific temporal realization of the behavioral calibration coefficients.
13. Evaluate the reference-relative outcome anchor.
14. Update π_θ through the native optimization procedure of the underlying MARL backbone.
15. end for

Output: behaviorally refined MARL policy π_θ.

The LLM is invoked only at phase boundaries and does not participate in real-time action selection. The final policy-update step in Algorithm F1 is an objective-level abstraction: MASAC, MAPPO, HAPPO, and MADPO retain their native optimization procedures rather than sharing an identical actor or critic loss.

The temporal realization of the behavioral preference is also backbone- and coordinate-dependent. Counteraction admits a direct step-level realization from executed joint actions, whereas dominant-agent switching and temporal action variation depend on temporal history and are evaluated from completed trajectory segments. Accordingly, when an exact temporal decomposition is unavailable, LOCM uses the corresponding step-wise or episode-level surrogate realization rather than assuming a universal exact decomposition.

MASAC implementation. For the complete MASAC-LOCM training path, the behavioral preference is realized through a mixed temporal implementation rather than by explicitly evaluating a single trajectory-level C_k at every environment step. At each step, the counteraction component contributes

$$r_{\mathrm{ctr},t} = -\lambda_{\mathrm{ctr}} \Psi_{\mathrm{ctr},t},$$

where

$$\Psi_{\mathrm{ctr},t} = \frac{\sum_i |a_{i,t}| - |\sum_i a_{i,t}|}{\sum_i |a_{i,t}| + 10^{-6}}.$$

At episode termination, the switching and temporal-variation components contribute

$$r_{\mathrm{terminal}} = -\lambda_{\mathrm{sw}} \Psi_{\mathrm{sw}} - \lambda_{\mathrm{var}} \Psi_{\mathrm{var}}.$$

The guided-phase training reward is then implemented as

$$r_t^{\mathrm{train}} = r_t^{\mathrm{structure}} - r_t^{\mathrm{wall}} - r_t^{\mathrm{relative\text{-}anchor}}.$$

During the guided phase, the raw environmental reward is retained separately as the environmental return used for evaluation and logging, but is not directly added to r_t^{train}. The resulting training reward is stored with the corresponding transitions in each agent's replay buffer and subsequently enters the standard SAC critic target and native SAC actor–critic updates.

Sections 3.3–3.4 introduce a reference-relative violation measure and an analytical penalized surrogate for task-prioritized behavioral refinement, with the latter analyzed in Appendix B. In the power-distribution implementation, the corresponding task-preservation mechanism is operationalized through a reference-relative physical acceptance gate that evaluates task-relevant quantities, including voltage deviation and network loss, against prescribed tolerances. Thus, the analytical surrogate and the implementation-level gate serve the same task-prioritization role without being mathematically identical, while the underlying MARL backbones retain their native optimization procedures.

F.2 LLM Semantic Interface

F.2.1 System Prompt

The semantic interface is designed to analyze execution behavior rather than to generate actions or directly construct numerical rewards. A condensed version of the system instruction used by LOCM is shown below.

```
System prompt

You are an expert in complex-system dynamics and
multi-agent coordination.

Analyze the behavioral trajectories of a multi-agent
control system.
```

```
Do not directly convert physical task metrics into
reward terms.

Treat task feasibility and safety as hard constraints.

Any recommended behavioral refinement must remain
within acceptable task-performance bounds.

If the current execution behavior becomes more regular
while task performance clearly deteriorates, do not
continue strengthening that behavioral pattern. Instead,
redirect the next phase toward alternative coordination
structures.

Return the analysis using the required structured JSON
format.
```

The prompt therefore assigns the LLM a diagnostic role. Task performance is supplied as context and constraint information, while the semantic output focuses on execution structure.

F.2.2 User Prompt Structure

At the end of phase k, the user message is assembled from aggregated trajectory statistics rather than raw trajectory arrays. Its general structure is:

```
PHASE {phase_index} ANALYSIS

Phase-level task summary:
 environmental return: {return_statistics}
 task deviation metric: {task_metric_1}
 secondary physical metric: {task_metric_2}
 reference-relative change: {reference_change}

Phase-level behavioral summary:
 action-sign counteraction: {counteraction_statistics}
 dominant-agent switching: {switching_statistics}
 temporal action variation: {variation_statistics}
 control burden / participation: {burden_statistics}
 convergence characteristics: {convergence_statistics}

Representative trajectories:
 {representative_trajectory_summaries}

Boundary or transition trajectories:
 {boundary_trajectory_summaries}

Recurring execution patterns:
 {recurring_bad_patterns}

Emerging execution patterns:
 {weak_good_clues}

Previous-phase comparison:
 {previous_phase_summary}

Historical phase information:
 {historical_context}

Retrieved similar phase cases:
```

```
 {retrieved_phase_context}

Return a structured semantic diagnosis using the
required JSON schema.
```

The prompt contains task metrics because behavioral refinement is conditioned on maintaining acceptable task performance. However, the prompt does not request real-time control actions and does not ask the LLM to directly output the numerical coefficients used by the MARL learning objective.

Representative and boundary trajectories are supplied in summarized form. This limits prompt size while retaining information about temporal execution patterns, regime transitions, and recurring multi-agent interactions. Previous-phase and historical information provide slow-timescale context for interpreting whether an observed behavioral pattern is persistent, emerging, or transient.

F.3 Structured Semantic Output and Numerical Interface

The LLM returns a structured semantic portrait rather than a numerical reward vector. The portrait contains phase-level diagnostic information, qualitative behavioral priorities, uncertainty information, and an optional action-style description.

A compact representation of the output structure is:

```
{
 "phase_stage": "...",
 "stability_potential": 0.0,
 "key_observations": ["...", "..."],
 "behavioral_focus": "...",
 "counteraction_suppression": "...",
 "role_switch_suppression": "...",
 "temporal_consistency_encouragement": "...",
 "burden_evolution_preference": "...",
 "participation_preference": "...",
 "phase_action_style": {
   "overall_intensity": "...",
   "collectiveness": "..."
 },
 "patterns_to_strengthen": ["...", "..."],
 "patterns_to_suppress": ["...", "..."],
 "uncertainty_areas": ["...", "..."],
 "reasoning": "..."
}
```

The structured output summarizes the current behavioral regime, qualitative behavioral priorities, uncertainty information, and phase-level action-style preferences. These semantic fields are subsequently converted into a numerical supervision target for phase-level calibration, as described in Section F.3.1.

F.3.1 Semantic-to-Numerical Adaptation

The structured semantic portrait is not used directly as a numerical learning signal. Instead, it is converted into a five-dimensional numerical supervision target,

$$y_k = (\lambda_{\text{ctr}}, \lambda_{\text{sw}}, \lambda_{\text{var}}, s_{\text{int}}, \rho_{\text{pull}}) = h(p_k).$$

For the three behavioral coordinates, the semantic levels share the mapping

$$L(\texttt{off}, \texttt{weak}, \texttt{medium}, \texttt{strong}, \texttt{very_strong}) = (0, 0.05, 0.10, 0.20, 0.35).$$

The corresponding targets are

$$\lambda_{\text{ctr}} = L(p_k.\texttt{counteraction_suppression}),$$

$$\lambda_{\text{sw}} = L(p_k.\texttt{role_switch_suppression}),$$

and

$$\lambda_{\text{var}} = L(p_k.\texttt{temporal_consistency_encouragement}).$$

The phase-level behavioral focus further modifies these targets. For `reduce_counteraction`, λ_{ctr} is multiplied by 1.5. For `stabilize_roles`, λ_{sw} and λ_{var} are multiplied by 1.3 and 1.2, respectively.

The action-intensity target is determined by

$$S(\texttt{very_conservative}) = 0.85, \quad S(\texttt{conservative}) = 0.92,$$
$$S(\texttt{slightly_conservative}) = 0.97, \quad S(\texttt{moderate}) = 1.00,$$
$$S(\texttt{slightly_aggressive}) = 1.03, \quad S(\texttt{aggressive}) = 1.08.$$

such that

$$s_{\text{int}} = S(p_k.\texttt{phase_action_style.overall_intensity}).$$

The collective-pull target is

$$P(\texttt{weak}, \texttt{medium}, \texttt{strong}) = (0, 0.02, 0.03),$$

with

$$\rho_{\text{pull}} = P(p_k.\texttt{phase_action_style.collectiveness}).$$

The trainable MLP adapter receives the 15-dimensional phase-level feature vector x_k, comprising current phase statistics, reference-relative physical ratios, and the previous control state, and produces

$$c_k = A_\phi(x_k).$$

At each phase boundary, the adapter is updated by one Adam optimization step against y_k and then remains fixed during the subsequent phase. Its realized output ranges are

$$\lambda_{\text{ctr}} \in (0, 0.60), \qquad \lambda_{\text{sw}} \in (0, 0.50), \qquad \lambda_{\text{var}} \in (0, 0.50),$$
$$s_{\text{int}} \in (0.90, 1.10), \qquad \rho_{\text{pull}} \in (0, 0.08).$$

In the implemented MASAC-LOCM path, the first three components are used directly as behavioral penalty coefficients rather than being simplex-normalized.

Action-style realization. The final two adapter outputs act on the actor-proposed joint action rather than entering the behavioral reward directly. Let $a = (a_1, \ldots, a_N)$ denote the actor-proposed joint action. Intensity scaling first gives

$$\tilde{a}_i = s_{\text{int}} a_i.$$

The mean scaled action is

$$\bar{a} = \frac{1}{N} \sum_{i=1}^{N} \tilde{a}_i.$$

Collective pull then moves each agent action toward the joint-action mean,

$$a'_i = \tilde{a}_i + \rho_{\text{pull}}(\bar{a} - \tilde{a}_i) = (1 - \rho_{\text{pull}})\tilde{a}_i + \rho_{\text{pull}}\bar{a}.$$

The action passed to the environment interface is

$$a_i^{\text{input}} = \text{clip}(a'_i, -1, 1).$$

The execution order is therefore actor proposal, intensity scaling, collective pull, clipping, and finally the environment-side constraint/noise interface. Behavioral statistics are computed from the executed actions returned through `info["exec_actions"]`, whereas the replay buffer stores the action-style-adjusted action array before environment-side execution noise is applied. This design separates semantic interpretation from numerical calibration: the LLM provides a structured behavioral diagnosis, the trainable adapter produces the corresponding numerical control variables, and the native MARL backbone remains responsible for policy optimization.

F.3.2 Reference-Relative Outcome Anchoring

Task preservation is handled independently of the semantic behavioral diagnosis. After establishment of the baseline reference, the implementation evaluates whether the current phase remains within an acceptable task-performance region.

For the power-distribution task, this operational anchor compares the phase-level average voltage deviation and average line loss against their corresponding baseline-reference values. Step-wise voltage-limit violations and SoC-related penalties remain part of the underlying task-side feasibility and penalty mechanisms, but they are not used as direct reference-relative criteria in the acceptance gate. Let $q_j(\pi)$ denote a physical evaluation quantity and q_j^{ref} its reference value. The practical acceptance condition can be represented abstractly as

$$q_j(\pi) \le \alpha_j q_j^{\mathrm{ref}}, \qquad j = 1, \dots, J. \tag{F5}$$

where α_j denotes the permitted reference-relative tolerance. For the reported power-distribution implementation, the two gate quantities are the phase-level average voltage deviation and average network loss, with

$$\alpha_{\mathrm{voltage}} = 1.05, \qquad \alpha_{\mathrm{loss}} = 1.10.$$

Accordingly, the implemented acceptance conditions are

$$\overline{\mathrm{VB}}_k \le 1.05\,\overline{\mathrm{VB}}_{\mathrm{ref}},$$

and

$$\overline{\mathrm{Loss}}_k \le 1.10\,\overline{\mathrm{Loss}}_{\mathrm{ref}}.$$

If either acceptance condition is violated, the implementation generates an automatic feedback message indicating that the current physical outcome is unacceptable. The next phase is then assigned the mode mixture

$$(\text{exploration}, \text{verification}, \text{exploitation}) = (0.7, 0.2, 0.1).$$

The violation feedback is appended to the next phase-level LLM prompt, instructing the subsequent phase not to continue reinforcing the current behavioral structure and instead to emphasize exploration and verification.

Equation (F5) operationalizes the same task-prioritization principle as Section 3.3, but is neither mathematically identical to the analytical penalty in Appendix B nor a guarantee of exact membership in Π_ϵ.

F.3.3 Fallback Mechanism

The LLM interface operates only at phase boundaries and is not required for real-time action execution. If the LLM service is unavailable, the API call fails, or a valid structured response cannot be parsed, LOCM applies a deterministic rule-based semantic analyzer to the same phase-level summary. The rule-based analyzer produces a structured semantic portrait compatible with the normal downstream numerical-adaptation interface, allowing training to continue without an online LLM response.

This failure-handling path is distinct from the Rule-diagnoser ablation reported in Section 4.5 and Appendix D.6. In full LOCM, the rule-based analyzer is invoked only as a fallback when the LLM interface is unavailable or invalid; in the Rule-diagnoser variant, deterministic rule-based diagnosis replaces the LLM-assisted semantic interface as the primary phase-level diagnostic mechanism. In both cases, semantic diagnosis remains separated from environment-step action selection and is processed through the downstream calibration pipeline before affecting subsequent policy updates.

F.3.4 Illustrative Semantic Diagnosis

Consider an illustrative phase in which task return remains close to the reference level, action-sign counteraction increases, dominant-agent switching remains moderate, and temporal action variation decreases.

A semantic diagnosis may take the form:

```
{
 "phase_stage": "behavioral_refinement",
 "stability_potential": 0.78,
 "key_observations": [
   "task performance remains within the reference region",
   "counteraction has increased despite lower temporal variation"
 ],
 "behavioral_focus": "reduce_counteraction",
 "counteraction_preference": "strong",
 "role_structure_preference": "medium",
 "temporal_consistency_preference": "maintain",
 "phase_action_style": {
   "overall_intensity": "moderate",
   "collectiveness": "medium"
 },
 "uncertainty_areas": [
   "whether the counteraction increase is transient"
 ]
}
```


This example is illustrative rather than an experimental observation. The structured semantic portrait is first converted into a numerical supervision target, which supervises the phase-boundary adapter update; the language model does not directly generate the numerical coefficients used by the MARL optimizer.