DISENTANGLE TO CORRECT: SEPARATING TEMPORAL STRUCTURE FROM NOISE IN MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Although the centralized training with decentralized execution (CTDE) paradigm has achieved notable success in multi-agent reinforcement learning, existing methods still struggle with robustness and long-horizon credit assignment, largely due to the entangled modeling of stable temporal structure and transient noise. To address this issue, we propose Structure–Noise Disentanglement (SND), a framework that explicitly factorizes agent representations into a structural branch for modeling long-term coordination structure and a noise branch for capturing instantaneous residuals. The structural branch employs Mamba, a selective state-space model, to efficiently capture reward-relevant long-range dependencies, while the noise branch is restricted to modeling only transient perturbations. Furthermore, we introduce an execution-time correction mechanism that leverages the learned structural representation to predict TD residuals, enabling lightweight and targeted value corrections without explicit planning. Experiments on the StarCraft Multi-Agent Challenge and Google Research Football benchmarks demonstrate superior sample efficiency and final performance compared to competitive baselines. Our code is available at: `https://anonymous.4open.science/r/CDN-EE4B`.

1 INTRODUCTION

Multi-agent reinforcement learning (MARL) aims to solve complex sequential decision-making problems through cooperation among multiple agents, and has demonstrated significant potential in domains such as robotic coordination, autonomous vehicle fleets, and multi-agent games (Hu et al., 2024a; Na et al., 2025). Most existing approaches adopt the centralized training with decentralized execution (CTDE) paradigm, which mitigates the non-stationarity inherent in multi-agent learning by sharing information during training while preserving scalability at execution time (Oliehoek et al., 2008; Foerster et al., 2018). Despite substantial progress in model architectures and representation capacity, current MARL methods still suffer from training instability, difficulty in long-term credit assignment, and limited robustness during execution, especially in environments with long temporal horizons, dense interactions, and significant noise.

To address these challenges, prior work has explored multiple directions. One line of research employs recurrent neural networks or Transformer-based architectures to encode agents' historical trajectories into a unified temporal representation, thereby enhancing expressiveness under partial observability (Hausknecht & Stone, 2015; Vaswani et al., 2017). Another line introduces world models or dynamics models to facilitate planning by predicting future states or rewards (Ha & Schmidhuber, 2018; Pitis et al., 2022). In addition, several studies investigate explainable reinforcement learning or causal modeling approaches to capture the relationships between agents' actions and long-term outcomes (Madumal et al., 2020; Grimbly et al., 2021). However, these methods typically encode reward-relevant coordination patterns and nuisance variations into a single temporal representation. When environmental noise is substantial or long-horizon credit assignment is required, such monolithic modeling can overfit short-term fluctuations, thereby weakening the representation of long-term decision structure and biasing value estimation during execution.

This issue is closely related to representation learning and disentanglement in RL, which study how to separate decision-relevant information from noisy or entangled inputs. Existing methods recover

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.

transition- and reward-preserving state beliefs from noisy observations (Cao et al., 2024), learn action-sufficient state representations for control (Huang et al., 2022), or construct task-independent state abstractions from learned dynamics (Wang et al., 2022c). In cooperative MARL, contrastive role representations support agent specialization and coordination (Hu et al., 2024b), while dynamics-based role learning connects agent roles with future behavior (Goel et al., 2025). These methods improve state representations or agent coordination, but they do not explicitly separate temporal structure from noise for execution-time value correction. When long-term coordination patterns and short-term fluctuations are encoded together, an agent may assign too much value to an attractive but fleeting opportunity. For example, in cooperative football, an agent may overvalue an immediate shot through a briefly open lane, even when passing to a teammate or waiting for a defender to move would create a better scoring opportunity. This motivates separating information that supports long-term coordination from noise that can distort local action values.

To this end, we propose Structure–Noise Disentanglement (SND), a framework that combines training-time representation disentanglement with execution-time value correction for CTDE-based cooperative MARL. SND separates reward-relevant temporal structure, including long-term coordination patterns, from nuisance variations and uses the structural component to correct local action values. During training, we instantiate the structural branch with Mamba to aggregate information across the agent's interaction history through input-dependent selection (Gu & Dao, 2024). The noise branch uses a feed-forward encoder without additional recurrent memory to model complementary behavioral variations. Structural behavior modeling and TD-residual prediction guide the structural branch toward decision- and value-relevant information, while residual behavior modeling and adversarial return supervision encourage the noise branch to capture nuisance variations. During execution, SND turns the learned structural representation into a lightweight correction signal. We train an action-conditioned predictor under team-level TD supervision to infer value discrepancies from each agent's local structural context. The TD residual measures the one-step bootstrap target minus the current joint value estimate and provides the supervision for this correction (Sutton et al., 1998). At execution time, only the top-k candidate Q-values are corrected using the predicted residuals, adjusting local action rankings without global information, additional environment interaction, or explicit future-trajectory prediction. This connects training-time disentanglement with targeted value correction during decentralized execution.

Our main contributions are as follows: (1) We propose Structure–Noise Disentanglement (SND), a unified framework that separates reward-relevant temporal structure from noise in each agent's representation during training, and uses the structural component to correct biased local Q-values during execution; (2) We instantiate this framework with disentanglement objectives and an action-conditioned TD-residual predictor, enabling lightweight top-k value correction without planning, future prediction, or additional interaction; (3) Experiments on Google Research Football and StarCraft II micromanagement tasks demonstrate the effectiveness of our approach, and ablation studies further validate the necessity of each core component.

2 Related Work

Multi-Agent Reinforcement Learning under CTDE. CTDE has become a dominant paradigm for cooperative MARL, as it mitigates training-time non-stationarity while preserving decentralized execution (Oliehoek et al., 2008; Foerster et al., 2018). Value decomposition methods, such as VDN, QMIX, QTRAN, and QPLEX, factorize the joint action-value function into agent-wise utilities under structural constraints (Sunehag et al., 2018; Son et al., 2019; Wang et al., 2020; Rashid et al., 2020). Recent extensions further improve expressiveness through richer mixing networks, role-based decomposition, or sub-team modeling (Wang et al., 2021; Phan et al., 2021; Zang et al., 2023). Despite their success, most CTDE methods still rely on monolithic agent representations, which may entangle long-term coordination structure with short-term fluctuations and lead to biased value estimates in noisy or long-horizon tasks.

Temporal Modeling and World Models in MARL. To handle partial observability and long-term dependencies, prior work has incorporated recurrent networks and attention-based architectures into MARL. RNN-based methods encode local histories as belief states (Hausknecht & Stone, 2015), while Transformer-based models capture long-range temporal interactions through self-attention (Vaswani et al., 2017). However, recurrent off-policy learning can suffer from instability

and biased gradients (Kapturowski et al., 2018). Another line of work learns world models or latent dynamics for planning and data augmentation (Ha & Schmidhuber, 2018; Janner et al., 2019), but accurate multi-step prediction is costly and difficult in decentralized settings. In contrast, our method does not predict full future trajectories; it extracts reward-relevant temporal structure for lightweight execution-time correction.

Representation Learning and Disentanglement in RL. Representation learning in RL aims to retain information needed for control while reducing sensitivity to irrelevant variations. Bisimulation-based methods learn task-relevant representations without reconstructing observations (Zhang et al., 2021), while Denoised MDPs factor latent dynamics according to controllability and reward relevance (Wang et al., 2022b). Related work separates state beliefs from observation noise through transition- and reward-preservation constraints (Cao et al., 2024). These approaches primarily construct state abstractions or latent world models for policy learning. SND instead separates reward-relevant temporal information from nuisance variations within each agent's history-informed representation, and uses the structural component to predict action-conditioned TD residuals for decentralized value correction.

3 PRELIMINARIES

We model cooperative multi-agent tasks as a decentralized partially observable Markov decision process (Dec-POMDP), defined by the tuple $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, R, \Omega, O, \gamma \rangle$. At each time step t, each agent $i \in \mathcal{N}$ receives a local observation $o_i^t \in \Omega$ sampled from the observation function $O(s^t, i)$, selects an action $a_i^t \in \mathcal{A}$, and forms a joint action $\mathbf{a}^t$. The environment then transitions to the next state $s^{t+1} \sim P(\cdot \mid s^t, \mathbf{a}^t)$ and emits a global reward $r^t = R(s^t, \mathbf{a}^t)$. The objective is to learn a joint policy π that maximizes the expected discounted return $\mathbb{E}_\pi[G]$, where $G = \sum_{t=0}^{\infty} \gamma^t r^t$.

4 STRUCTURE–NOISE REPRESENTATION FACTORIZATION

To guide the design of the disentanglement module, we distinguish reward-relevant temporal structure from noise in agent decision representations. Here, *temporal structure* refers to information across the interaction history that supports long-term coordination and value estimation, while *noise* denotes nuisance variations that help explain observed behavior without consistently informing long-term returns. This distinction provides a structural inductive bias for representation learning and loss construction.

Specifically, we learn two latent components from the agent's local decision context:

$$z_i^{s,t} = f_s(x_i^{1:t}), \qquad z_i^{n,t} = f_n(x_i^t). \tag{1}$$

Here, x_i^t is the history-informed shared input defined in Eq. (4), and $x_i^{1:t}$ denotes its sequence up to time t. The structural encoder f_s aggregates this sequence with additional temporal memory, while the noise encoder f_n processes the current shared input without maintaining an additional temporal state. The superscripts s and n denote structure and noise, respectively.

Based on this factorization, we formulate an auxiliary behavior model:

$$\ell_i^t = \underbrace{W_s z_i^{s,t}}_{\text{Structural logits}} + \underbrace{W_n z_i^{n,t}}_{\text{Noise logits}}, \qquad \pi_i^t(\cdot) = \mathrm{softmax}(\ell_i^t). \tag{2}$$

Here, W_s and W_n are learnable linear transformations. Both paths contribute to behavior reconstruction, while only the structural representation is reused for execution-time value correction.

We organize this factorization around three design principles:

Design Principle 1 (Reward-Relevance)**.** The structural representation $z_i^{s,t}$ is encouraged to retain reward-relevant temporal information for behavior modeling and TD-residual prediction. The noise representation $z_i^{n,t}$ explains complementary behavioral variations while its long-term return predictability is suppressed.

Design Principle 2 (Temporal Asymmetry)**.** Both branches receive history-informed context, but only the structural encoder maintains additional temporal memory for cross-time information aggregation. The noise encoder maps each shared input directly to its representation.

Design Principle 3 (Representational Decoupling)**.** During learning, the structural representation $z_i^{s,t}$ and the noise representation $z_i^{n,t}$ are encouraged to be orthogonal in latent space, while complementary learning objectives promote their distinct functional roles.

These principles translate the structure–noise distinction into architectural asymmetry and complementary learning objectives, directing reward-relevant temporal information toward the representation used for execution-time TD-residual correction.

5 Structure–Noise Disentanglement Module

To characterize the local decision context under partial observability, we build on the recurrent utility and role-representation modules of the underlying CTDE learner. Following ACORM (Hu et al., 2024b), each agent obtains a role embedding r_i^t from its local interaction history, encouraging agents with similar behavioral patterns to form nearby role representations.

The recurrent local-utility network then takes the current observation, previous action, and role embedding as input, and maintains a hidden belief state b_i^t that summarizes the agent's action–observation history:

$$\big(Q_i(\tau_i^t, \cdot),\, b_i^t\big) = F_Q\left([o_i^t; \tilde{a}_i^{t-1}; r_i^t], b_i^{t-1}\right). \tag{3}$$

We combine the belief state, role representation, and previous action to form the shared input to the two representation branches:

$$x_i^t = \left[\mathrm{sg}(b_i^t); \mathrm{sg}(r_i^t); \tilde{a}_i^{t-1}\right], \tag{4}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient. Both branches receive the base learner's decision context through detached belief-state and role-embedding inputs.

5.1 Structural Branch: Selective Temporal Encoder

The structural branch is designed to aggregate reward-relevant information across the agent's interaction history for long-term coordination and value correction. Unlike the noise branch, it maintains an additional temporal state to capture cross-time dependencies. To realize this temporal asymmetry while supporting efficient incremental inference, we adopt Mamba as the structural encoder (Gu & Dao, 2024).

Given the shared input x_i^t, we first compute

$$u_i^t = \mathrm{LN}\big(\mathrm{ReLU}(W_{\mathrm{in}} x_i^t)\big). \tag{5}$$

At its core, Mamba builds on a continuous-time state-space model,

$$\frac{dh_i(t)}{dt} = Ah_i(t) + Bu_i(t), \qquad y_i(t) = Ch_i(t), \tag{6}$$

where $h_i(t) \in \mathbb{R}^{d_{\mathrm{state}}}$ denotes the latent state.

A key distinction of Mamba is its input-dependent selection mechanism. Specifically, $(\Delta_i^t, B_i^t, C_i^t) = \Psi(u_i^t)$, where Ψ denotes the input-conditioned projections and $\Delta_i^t > 0$ is obtained through a softplus transformation. After discretization,

$$\begin{aligned} s_i^t &= \bar{A}_i^t s_i^{t-1} + \bar{B}_i^t u_i^t, \\ y_i^t &= C_i^t s_i^t + D \odot u_i^t, \end{aligned} \tag{7}$$

where $\bar{A}_i^t$ and $\bar{B}_i^t$ are the corresponding discretized state-transition and input operators.

The resulting structural representation is

$$z_i^{s,t} = W_o\big[y_i^t \odot \mathrm{SiLU}(W_g u_i^t)\big]. \tag{8}$$

During execution, the recurrent state is cached and updated incrementally, allowing $z_i^{s,t}$ to summarize the interaction history without reprocessing the full sequence.

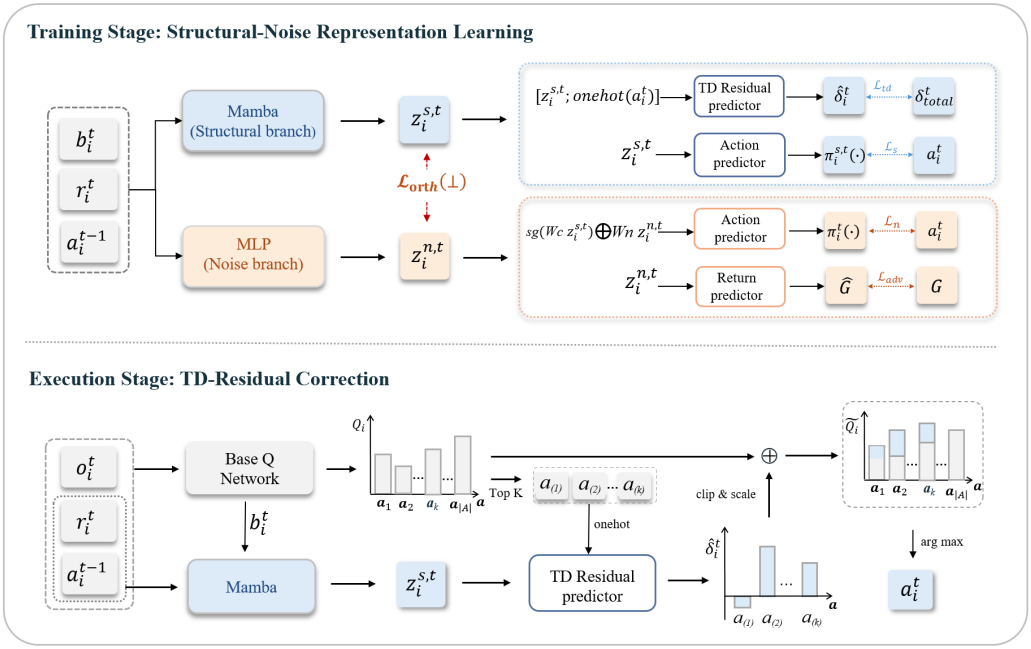

Figure 1: Overview of the proposed structure–noise disentanglement framework. The structural branch models reward-relevant temporal structure using a selective state-space model (Mamba), while the noise branch captures complementary behavioral variations through residual learning. During training, complementary objectives disentangle the two representations; during execution, the structural representation predicts action-conditioned TD residuals for lightweight correction of top-k candidate action values.

5.2 Noise Branch: MLP-based Residual Encoder

The noise branch complements the structural branch by using residual behavior modeling to capture nuisance variations at each decision step. We implement it as a feed-forward encoder that processes the current shared input without maintaining an additional temporal state:

$$z_i^{n,t} = f_n(x_i^t) = \mathrm{MLP}(x_i^t), \tag{9}$$

where f_n contains no recurrent states or inference caching mechanisms.

Both branches receive the same history-informed input x_i^t, including the belief state b_i^t. Their architectural asymmetry lies in how this context is processed: the structural branch maintains additional temporal memory, while the noise branch maps each input directly to a residual representation. The residual behavior modeling and adversarial return objectives in Section 5.3 further encourage the noise branch to explain complementary behavioral variations while suppressing return predictability.

5.3 Disentangled Representation Learning via Complementary Objectives

To learn the structure–noise separation without explicit labels, we design a set of complementary objectives that assign distinct functional roles to the two branches. The structural branch is encouraged to retain behavior- and value-relevant temporal information, while the noise branch models complementary behavioral variations and is discouraged from encoding long-term return information. Figure 1 illustrates the complete training and execution pipeline. During training, the structural representation is additionally used to predict the action-conditioned TD residual under team-level TD supervision. During execution, only the structural encoder and TD-residual predictor are retained to apply lightweight corrections to the top-k candidate action values.

(i) Structural Behavior Modeling Loss. To preserve information on the agent's primary decision patterns in the structural representation $z_i^{s,t}$, we employ a behavior cloning loss as the supervision:

$$\mathcal{L}_{\mathrm{s}} = \mathbb{E}\left[\sum_{i=1}^{N} \mathrm{CE}\big(\pi_i^{s,t}(\cdot), a_i^t\big)\right]. \tag{10}$$

where $\pi_i^{s,t}(\cdot) = \mathrm{softmax}(W_s z_i^{s,t})$ denotes the structural behavior distribution and a_i^t is the recorded action. This term encourages the structural branch to retain behavior-predictive information and counteracts representation collapse.

(ii) Noise Residual Modeling Loss with Stop-Gradient. To give the noise branch $z_i^{n,t}$ a complementary behavior-modeling role, we use a residual learning mechanism. Specifically, its output is added to the *stop-gradient* of the structural logits:

$$\mathcal{L}_{\mathrm{n}} = \mathbb{E}\left[\sum_{i=1}^{N} \mathrm{CE}\left(\mathrm{softmax}\big(\mathrm{sg}(W_s z_i^{s,t}) + W_n z_i^{n,t}\big), a_i^t\right)\right]. \tag{11}$$

where $\pi_i^t(\cdot)$ denotes the residual-augmented behavior distribution defined in Eq. (2), and $\mathrm{sg}(\cdot)$ is the stop-gradient operator. Since stop-gradient does not alter the forward values, the distribution used above is identical to $\pi_i^t(\cdot)$ in the forward pass, while gradients from $\mathcal{L}_{\mathrm{n}}$ are blocked from the structural logits. This trains the noise branch to explain behavioral variations not captured by the structural logits. The adversarial objective below further discourages these residual features from retaining long-term return information.

(iii) Representation Separation via Orthogonality Regularization. To encourage directional separation between the two branches, we minimize the squared cosine similarity of their representations at each decision step:

$$\mathcal{L}_{\mathrm{orth}} = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} \cos^2(z_i^{s,t}, z_i^{n,t})\right]. \tag{12}$$

By penalizing aligned representations, this term provides a geometric separation bias that complements the functional specialization encouraged by the branch-specific objectives.

(iv) Adversarial Return-Prediction Loss. To discourage the noise representation $z_i^{n,t}$ from carrying information predictive of the long-term return G^t, we introduce a discriminator g_{adv} trained to predict G^t from $z_i^{n,t}$, while the noise encoder is updated via a *Gradient Reversal Layer* (GRL) to maximize this prediction error:

$$\mathcal{L}_{\mathrm{adv}} = \mathbb{E}\left[\sum_{i=1}^{N} \left\| g_{\mathrm{adv}}(\mathrm{GRL}_\lambda(z_i^{n,t})) - G^t \right\|_2^2\right]. \tag{13}$$

These opposing updates suppress return-predictive information in the noise branch, complementing residual behavior modeling to promote its specialization in nuisance variations.

(v) TD-Residual Prediction for Value Correction. The structural representation is additionally trained to predict the TD discrepancy of the underlying value-decomposition learner. Let

$$y^t = r^t + \gamma(1-d^t) Q_{\mathrm{tot}}^{\mathrm{tgt}}\left(\boldsymbol{\tau}^{t+1}, \mathbf{a}^{*,t+1}, s^{t+1}\right) \tag{14}$$

denote the one-step bootstrap target, where $\mathbf{a}^{*,t+1}$ is formed by the greedy available actions of the online local utility networks.

We define the team-level TD residual as

$$\delta_{\mathrm{tot}}^t = y^t - Q_{\mathrm{tot}}^{\mathrm{eval}}\left(\boldsymbol{\tau}^t, \mathbf{a}^t, s^t\right), \tag{15}$$

and clip it to $[-c, c]$ when constructing the auxiliary target. Each agent predicts this shared discrepancy conditioned on its own structural representation and executed action:

$$\hat{\delta}_i^t = g_{\mathrm{td}}\left([z_i^{s,t}; \mathrm{onehot}(a_i^t)]\right). \tag{16}$$

The corresponding prediction objective is

$$\mathcal{L}_{\mathrm{td}} = \mathbb{E}\left[\sum_{i=1}^{N}\left(\hat{\delta}_i^t - \mathrm{clip}\left(\delta_{\mathrm{tot}}^t, -c, c\right)\right)^2\right]. \tag{17}$$

The same team-level residual is used as supervision for all agents, while the action-conditioned predictor learns how this shared value discrepancy manifests under each agent's local structural context. We therefore treat δ_{tot}^t as an auxiliary correction signal rather than an identified individual advantage.

Total Optimization and Stability. The disentanglement objective combines the two behavior-modeling losses with the auxiliary regularizers and TD-residual supervision:

$$\mathcal{L}_{\mathrm{dis}} = \mathcal{L}_{\mathrm{s}} + \mathcal{L}_{\mathrm{n}} + \lambda_o \mathcal{L}_{\mathrm{orth}} + \lambda_a \mathcal{L}_{\mathrm{adv}} + \lambda_t \mathcal{L}_{\mathrm{td}}. \tag{18}$$

The two behavior-modeling losses have unit weights, while λ_o, λ_a, and λ_t control the auxiliary terms. The belief-state and role-embedding inputs are detached, while $\mathcal{L}_{\mathrm{td}}$ retains the computation graph of the team-level TD residual. This loss jointly trains the structural encoder and residual predictor and contributes gradients to the base utility, mixing, and role-representation networks.

Collectively, the proposed objectives translate our structure–noise design principles into learning signals: $\mathcal{L}_{\mathrm{s}}$ and $\mathcal{L}_{\mathrm{td}}$ train the structural representation to support behavior prediction and value correction; $\mathcal{L}_{\mathrm{n}}$ provides complementary behavior supervision, while $\mathcal{L}_{\mathrm{adv}}$ suppresses return predictability in the noise branch; and $\mathcal{L}_{\mathrm{orth}}$ encourages directional separation between the two representations in latent space.

5.4 TD-Residual-Guided Action Correction

Standard decentralized execution selects actions greedily according to the estimated local utilities. Under partial observability and non-stationary coordination, however, these utilities may suffer from systematic value bias. To mitigate this issue, we augment the local utilities with a lightweight, learnable correction derived from the structural representation.

An effective correction signal should reflect value-estimation discrepancies while remaining computable from local information during decentralized execution. We therefore use the team-level TD residual defined in Section 5.3 as supervision. This residual measures the one-step bootstrap target minus the current joint value estimate. By conditioning its prediction on an agent's local structural representation and a candidate action, SND learns a signed correction signal that can be evaluated without centralized information at execution time.

For each available action a, the correction head predicts

$$\hat{\delta}_i^t(a) = g_{\mathrm{td}}\left([z_i^{s,t}; \mathrm{onehot}(a)]\right). \tag{19}$$

Rather than evaluating this correction for the complete action set, we first select the k highest-valued actions under the base utility:

$$\mathcal{C}_{i,k}^t = \mathrm{TopK}_{a \in \mathcal{A}_{i,\mathrm{av}}^t} Q_i(\tau_i^t, a). \tag{20}$$

The corrected utility is

$$\tilde{Q}_{i,k}^t(a) = Q_i(\tau_i^t, a) + \alpha\, \mathbf{1}\left[a \in \mathcal{C}_{i,k}^t\right] \mathrm{clip}\left(\hat{\delta}_i^t(a), -c, c\right), \tag{21}$$

where α controls the correction strength and c limits the effect of extreme residual predictions.

The final decision is made over the complete set of available actions:

$$a_i^t = \arg\max_{a \in \mathcal{A}_{i,\mathrm{av}}^t} \tilde{Q}_{i,k}^t(a). \tag{22}$$

Thus, top-k determines which actions are *corrected*, rather than restricting which actions can ultimately be selected. The structural representation is computed once per step, and only the lightweight residual head is evaluated for the selected candidates.

5.5 Integration with Value Decomposition

SND is built on top of the original value-decomposition learner rather than replacing its local utility function. The base utilities are aggregated through the monotonic mixing network:

$$Q_{\text{tot}}(\boldsymbol{\tau}^t, \mathbf{a}^t, s^t) = f_{\text{mix}}\left(Q_1(\tau_1^t, a_1^t), \ldots, Q_N(\tau_N^t, a_N^t); s^t\right). \tag{23}$$

The primary TD objective for value learning is:

$$\mathcal{L}_{\text{RL}} = \mathbb{E}_{\mathcal{D}}\left[\left(Q_{\text{tot}}^{\text{eval}}(\boldsymbol{\tau}^t, \mathbf{a}^t, s^t) - \text{sg}(y^t)\right)^2\right], \tag{24}$$

where y^t is the bootstrap target defined in Eq. (14).

Combining the primary RL objective with the disentanglement objective gives

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{RL}} + \lambda_{\text{dis}}\mathcal{L}_{\text{dis}}, \tag{25}$$

where λ_{dis} controls the overall weight of the disentanglement objective.

SND therefore preserves the base value-decomposition architecture and augments it with a disentanglement-and-correction pathway. During execution, the noise branch and training-only auxiliary heads are removed; only the structural encoder and TD-residual prediction head are retained to correct the base local utilities.

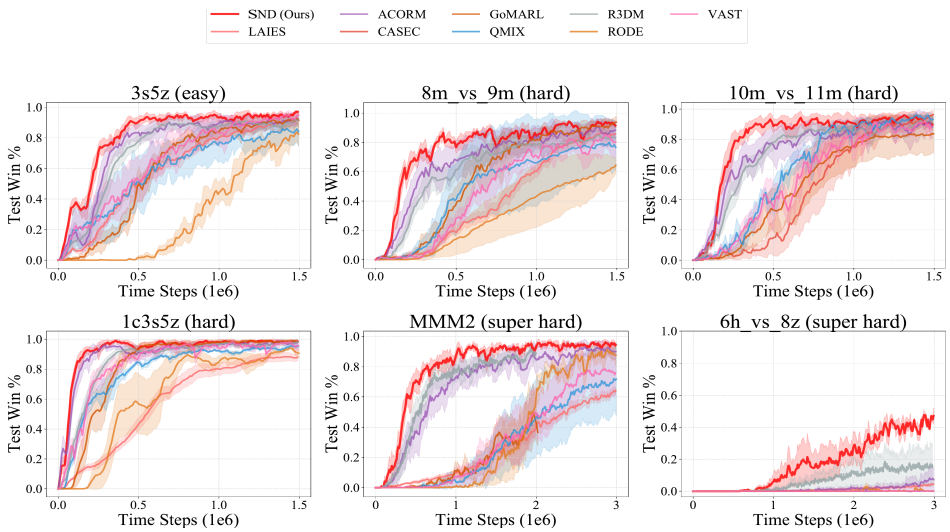

Figure 2: Comparison of SND against baseline algorithms on six SMAC maps.

6 Experiments

Experiments are conducted on two widely used cooperative MARL benchmarks: StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) and Google Research Football (GRF) (Kurach et al., 2020). The main benchmark results are averaged over five random seeds, with mean test win rates shown as solid curves and 95% bootstrapped confidence intervals as shaded regions. Additional analyses in the appendix report their seed counts separately.

Baselines. In addition to QMIX (Rashid et al., 2020), we compare SND with several representative recent baselines, including role-based methods (RODE (Wang et al., 2021), ACORM (Hu et al., 2024b), R3DM (Goel et al., 2025)), the causality-inspired LAIES (Liu et al., 2023), and structure-aware coordination methods (CASEC (Wang et al., 2022a), VAST (Phan et al., 2021), GoMARL (Zang et al., 2023)).

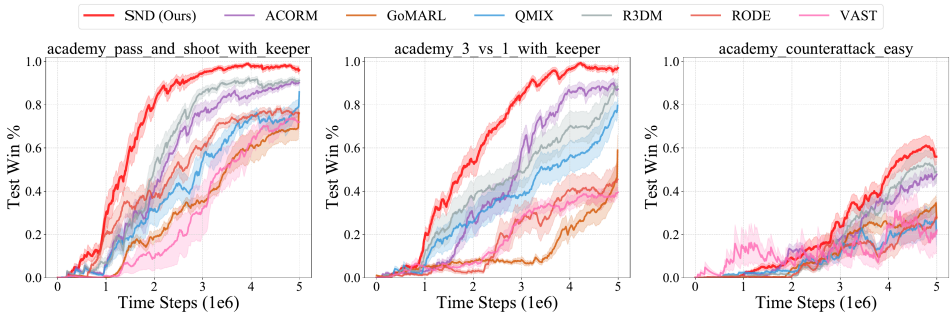

Figure 3: Comparison of SND against baseline algorithms on GRF.

Performance on SMAC Figure 2 compares **SND** with representative baselines on six SMAC scenarios. On the simple *3s5z* task, most methods reach high win rates, while SND learns faster in the early stage. On harder tasks such as *8m_vs_9m* and *10m_vs_11m*, SND shows more stable and efficient learning, whereas several baselines converge slowly or exhibit larger fluctuations. On the super-hard scenarios, SND reaches strong performance earlier on *MMM2* and achieves clearly higher win rates on *6h_vs_8z*, where most baselines struggle. These results demonstrate that SND effectively improves sample efficiency and mitigates long-horizon value bias.

Performance on Google Research Football. We further evaluate SND on three Google Research Football academy scenarios. As shown in Fig. 3, SND consistently outperforms all baselines, with faster convergence on the two offensive coordination tasks and the best final performance on *academy_counterattack_easy*. Since GRF involves longer-horizon coordination and delayed scoring feedback, these results suggest that SND generalizes beyond SMAC micromanagement and improves robust decentralized action ranking in temporally extended cooperative tasks.

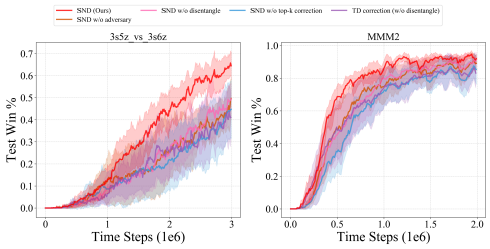

Figure 4: Ablation studies of SND. TD correction (w/o disentangle) applies TD-residual correction to a monolithic encoder.

6.1 Ablation Study

To evaluate each component in SND, we conduct ablation studies. As shown in Figure 4, the full SND consistently outperforms all variants. Removing top-k correction degrades convergence and final performance, while TD correction without disentanglement remains clearly inferior to SND, indicating that correction alone is insufficient. Meanwhile, SND without top-k correction remains competitive, showing that structure–noise disentanglement itself provides independent benefits. The drop of SND without adversarial supervision further supports the importance of suppressing return-related information in the noise branch. Further representation and correction diagnostics are provided in Appendix D.

7 Conclusions

This paper addresses long-horizon instability and execution-time value bias in partially observable multi-agent settings by proposing SND, a framework that disentangles representations during training and corrects decisions during execution. Our key insight is that reward-relevant temporal structure and nuisance variations play different roles in decentralized decision-making; separating them enables the learned structure to support targeted value correction. Guided by this insight, we disentangle temporal structure and noise via complementary objectives during training and apply lightweight, action-conditioned TD-residual–based corrections during execution, while preserving the original CTDE framework and high computational efficiency. Its adaptability in highly non-stationary or adversarial settings warrants further investigation.

AI USE STATEMENT

Generative AI tools were used to aid and polish the writing of the manuscript, including improvements to clarity, grammar, and phrasing. They were also used for retrieval and discovery of relevant literature. All AI-assisted writing and retrieved references were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper.

REFERENCES

Haiyao Cao, Zhen Zhang, Panpan Cai, Yuhang Liu, Jinan Zou, Ehsan Abbasnejad, Biwei Huang, Mingming Gong, Anton van den Hengel, and Javen Qinfeng Shi. Rethinking state disentanglement in causal reinforcement learning. *arXiv preprint arXiv:2408.13498*, 2024.

Jakob 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, 2018.

Harsh Goel, Mohammad Omama, Behdad Chalaki, Vaishnav Tadiparthi, Ehsan Moradi Pari, and Sandeep Chinchali. R3dm: Enabling role discovery and diversity through dynamics models in multi-agent reinforcement learning. *arXiv preprint arXiv:2505.24265*, 2025.

St John Grimbly, Jonathan Shock, and Arnu Pretorius. Causal multi-agent reinforcement learning: Review and open problems. In *Cooperative AI Workshop, Advances in Neural Information Processing Systems*, 2021.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Matthew J Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI fall symposia*, volume 45, pp. 141, 2015.

Yue Hu, Yuzhu Cai, Yaxin Du, Xinyu Zhu, Xiangrui Liu, Zijie Yu, Yuchen Hou, Shuo Tang, and Siheng Chen. Self-evolving multi-agent collaboration networks for software development. *arXiv preprint arXiv:2410.16946*, 2024a.

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 (ICLR)*, 2024b.

Biwei Huang, Chaochao Lu, Liu Leqi, José Miguel Hernández-Lobato, Clark Glymour, Bernhard Schölkopf, and Kun Zhang. Action-sufficient state representation learning for control with structural constraints. In *International Conference on Machine Learning*, pp. 9260–9279. PMLR, 2022.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

Steven Kapturowski, Georg Ostrovski, John Quan, Remi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *International conference on learning representations*, 2018.

Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 4501–4510, 2020.

Boyin Liu, Zhiqiang Pu, Yi Pan, Jianqiang Yi, Yanyan Liang, and Du Zhang. Lazy agents: A new perspective on solving sparse reward problem in multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 21937–21950. PMLR, 2023.

Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. Explainable reinforcement learning through a causal lens. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 2493–2500, 2020.

Hyungho Na, Kwanghyeon Lee, Sumin Lee, and Il-Chul Moon. Trajectory-class-aware multi-agent reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

Thomy Phan, Fabian Ritz, Lenz Belzner, Philipp Altmann, Thomas Gabor, and Claudia Linnhoff-Popien. Vast: Value function factorization with variable agent sub-teams. *Advances in neural information processing systems*, 34:24018–24032, 2021.

Silviu Pitis, Elliot Creager, Ajay Mandlekar, and Animesh Garg. Mocoda: Model-based counterfactual data augmentation. *Advances in Neural Information Processing Systems*, 35:18143–18156, 2022.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(1):1–51, 2020.

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, Woo Jae Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. *International Conference on Machine Learning (ICML)*, pp. 5887–5896, 2019.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning. *Proceedings of the Adaptive and Learning Agents Workshop (AAMAS)*, 2018.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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

Jianhao Wang, Zhizhou Ren, Tonghan Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations (ICLR)*, 2020.

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

Tonghan Wang, Liang Zeng, Weijun Dong, Qianlan Yang, Yang Yu, and Chongjie Zhang. Context-aware sparse deep coordination graphs. In *International Conference on Learning Representations*, 2022a.

Tongzhou Wang, Simon Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised MDPs: Learning world models better than the world itself. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22591–22612. PMLR, 2022b.

Zizhao Wang, Xuesu Xiao, Zifan Xu, Yuke Zhu, and Peter Stone. Causal dynamics learning for task-independent state abstraction. In *International Conference on Machine Learning*, pp. 23151–23180. PMLR, 2022c.

Yifan Zang, Jinmin He, Kai Li, Haobo Fu, Qiang Fu, Junliang Xing, and Jian Cheng. Automatic grouping for efficient cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 36:46105–46121, 2023.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021.

A Bias-Aware Analysis of TD-Residual Correction

We analyze how action-dependent estimation bias and the learned correction affect local action ranking. Consider a single agent at a fixed local history τ with a nonempty finite available-action set $\mathcal{A}$; agent and time indices are omitted. Let $\mathcal{C}_k$ contain the $\min(k, |\mathcal{A}|)$ highest-ranked actions under the base utility Q, where $k \geq 1$. Define

$$\bar{\delta}_k(a) = \mathbf{1}[a \in \mathcal{C}_k] \operatorname{clip}(\hat{\delta}(a), -c, c), \qquad \widetilde{Q}_k(a) = Q(a) + \alpha \bar{\delta}_k(a), \tag{26}$$

where $\alpha \geq 0$ and $c \geq 0$. The final action maximizes $\widetilde{Q}_k$ over all available actions, with deterministic tie-breaking.

Let $Q^\dagger$ be a fixed reference utility conditional on τ, with a unique preferred action $a^\dagger$ and gaps

$$\Delta^\dagger(a) = Q^\dagger(a^\dagger) - Q^\dagger(a) > 0, \qquad a \neq a^\dagger. \tag{27}$$

Write $Q(a) = Q^\dagger(a) + \varepsilon(a)$. Expectations and variances below are over a specified distribution of learned estimators, such as independent training runs evaluated at the same history. The candidate set and correction predictions may depend on the base estimation errors.

Proposition 1 (Bias-aware action-ranking bound)**.** For each competing action $a \neq a^\dagger$, define

$$\begin{aligned} b_a &= \mathbb{E}[\varepsilon(a) - \varepsilon(a^\dagger) \mid \tau], \\ m_a &= \mathbb{E}[\bar{\delta}_k(a^\dagger) - \bar{\delta}_k(a) \mid \tau], \\ D_a &= \widetilde{Q}_k(a^\dagger) - \widetilde{Q}_k(a), \\ M_a &= \mathbb{E}[D_a \mid \tau] = \Delta^\dagger(a) - b_a + \alpha m_a, \\ \sigma_a^2 &= \operatorname{Var}(D_a \mid \tau) < \infty. \end{aligned} \tag{28}$$

The quantities b_a and m_a may have either sign. Whenever $M_a > 0$,

$$\mathbb{P}\Big[\widetilde{Q}_k(a) \geq \widetilde{Q}_k(a^\dagger) \mid \tau\Big] \leq \frac{\sigma_a^2}{\sigma_a^2 + M_a^2}. \tag{29}$$

If $M_a > 0$ for every $a \neq a^\dagger$, then

$$\mathbb{P}\left[\arg\max_{a \in \mathcal{A}} \widetilde{Q}_k(a) = a^\dagger \mid \tau\right] \geq \max\left\{0,\, 1 - \sum_{a \neq a^\dagger} \frac{\sigma_a^2}{\sigma_a^2 + M_a^2}\right\}. \tag{30}$$

Proof. Expanding D_a gives its conditional mean $M_a = \Delta^\dagger(a) - b_a + \alpha m_a$. Let $Z_a = D_a - M_a$, so that $\mathbb{E}[Z_a \mid \tau] = 0$ and $\mathbb{E}[Z_a^2 \mid \tau] = \sigma_a^2$. For any $u \geq 0$, the event $Z_a \leq -M_a$ implies $(Z_a - u)^2 \geq (M_a + u)^2$. Hence

$$\mathbb{P}(D_a \leq 0 \mid \tau) \leq \frac{\sigma_a^2 + u^2}{(M_a + u)^2}.$$

Choosing $u = \sigma_a^2 / M_a$ yields Eq. (29), the one-sided Chebyshev–Cantelli bound. Failure to select $a^\dagger$ requires $D_a \leq 0$ for at least one competing action. A union bound gives Eq. (30). □

B Hyperparameter Sensitivity

We further evaluate the sensitivity of SND to two key hyperparameters: the correction strength α and the TD-residual clipping threshold c. The results show that SND remains robust over a moderate range of hyperparameter choices.

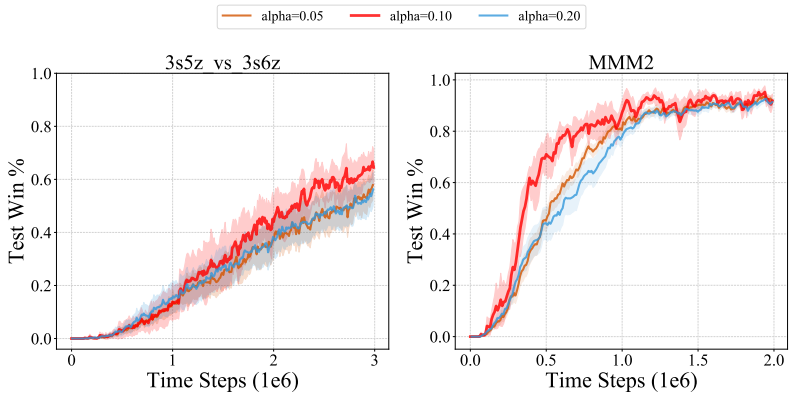

Figure 5: Effect of the correction strength α.

B.1 Effect of Correction Strength α

As shown in Figure 5, SND maintains strong and stable performance across all tested values of α, indicating that the method is not overly sensitive to the correction strength. The coefficient α controls the magnitude of the TD-residual correction added to the original decentralized utility. Although different values lead to slightly different learning dynamics, all tested settings remain effective. Specifically, $\alpha = 0.05$ applies a conservative correction and therefore converges slightly more slowly, while $\alpha = 0.20$ introduces a stronger correction and may cause mild training fluctuations. The intermediate value $\alpha = 0.10$ achieves a favorable balance between correction effectiveness and stability. Overall, the differences across these settings are moderate, suggesting that SND benefits from the correction mechanism without relying on delicate tuning of α.

B.2 Effect of Clipping Threshold c

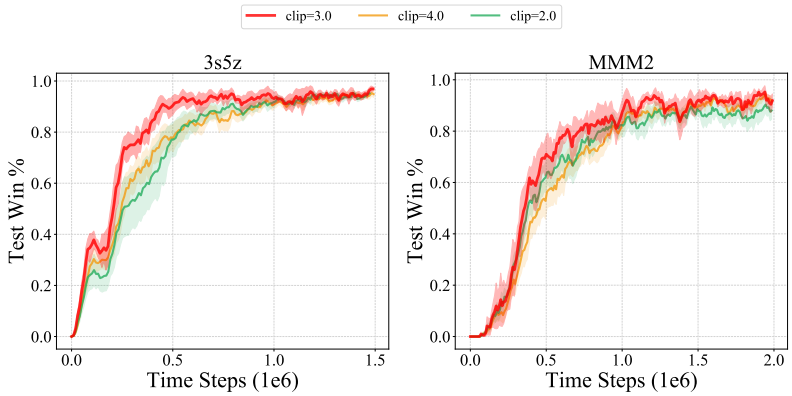

Figure 6: Effect of the TD-residual clipping threshold c.

Figure 6 shows the effect of the TD-residual clipping threshold c. SND achieves stable performance across all tested thresholds, indicating that the proposed correction mechanism is reasonably robust to this hyperparameter. The threshold c controls how aggressively large predicted residuals are clipped before being used for execution-time correction. A smaller threshold, such as $c = 2.0$, provides stronger protection against extreme correction values, but may also suppress useful correction signals. A larger threshold, such as $c = 4.0$, preserves more residual information but can retain larger and potentially less reliable corrections. The setting $c = 3.0$ provides a favorable balance between these effects. Nevertheless, all tested values yield competitive final performance, suggesting that SND remains effective over a moderate range of c.

C Residual Prediction Quality and Alignment

Proposition 1 shows that the ranking-error bound becomes tighter when the learned correction increases the corrected action margin. Motivated by this result, we empirically examine the directional alignment between predicted residual differences and estimated action-value gaps, as well as residual prediction accuracy under different decision difficulty levels. We further compare SND with a non-disentangled encoder to evaluate whether structure–noise disentanglement improves residual alignment.

Table 1: Residual prediction quality and alignment diagnostics. The disentangled encoder achieves stronger sign correlation and higher prediction accuracy than the non-disentangled encoder across both maps and difficulty levels.

Encoder	Map	Sign correlation	Hard acc.	Medium acc.	Easy acc.
Disentangled	*3s5z*	0.8293	0.7064	0.7867	0.8409
	MMM2	0.8890	0.6825	0.7415	0.8625
Non-disentangled	*3s5z*	0.7685	0.6412	0.7248	0.8031
	MMM2	0.8217	0.6189	0.6897	0.8264

As shown in Table 1, the predicted residuals are strongly aligned with the estimated value-gap direction. With the disentangled encoder, the sign correlation reaches 0.8293 on *3s5z* and 0.8890 on *MMM2*, indicating that the learned TD residual provides a directionally meaningful correction signal. Prediction accuracy is highest on easier decisions and decreases on harder cases, but remains substantially above chance even in the hard regime.

The non-disentangled encoder shows weaker alignment across all reported metrics. Both sign correlation and prediction accuracy decrease on the two maps, with the most pronounced gap appearing on hard decisions. These results suggest that disentanglement is particularly useful when reliable value correction requires separating reward-relevant temporal structure from nuisance variations. Overall, these diagnostics support structure–noise disentanglement as a means of improving residual alignment for execution-time value correction.

D Additional Empirical Analyses

D.1 Simulator-Level Interventions

We examine how the two representation branches respond to reward-relevant environmental changes on *MMM2*. We use two independently trained SND models and 24 paired episodes per training seed. The unmodified, intervened, and sham environments share the same environment seed and follow identical actions before intervention. The maximum pre-intervention observation discrepancy and the number of action mismatches are both zero.

We multiply enemy units' current health by 0.5 or 1.5 through the SC2 simulator. The sham condition follows the same procedure without changing health. Ridge regressors are fitted separately in the two learned representation spaces and evaluated using the same episode-grouped cross-validation protocol. They predict changes in future environmental return from changes in either the structural representation z^s or the noise representation z^n.

Table 2: Predicting intervention-induced return changes on *MMM2*. Entries are Pearson correlations between out-of-fold probe predictions and observed return changes. Each training seed uses 24 paired episodes.

Training seed	Structural branch	Noise branch
Seed 1	0.780	0.250
Seed 2	0.820	0.220

As shown in Table 2, predictions from the structural branch have higher correlations with the observed return changes for both training seeds. The arithmetic mean of the two seed-wise correlations is

0.800 for the structural branch and 0.235 for the noise branch. Under these health interventions, return changes are more strongly associated with the linear predictions from structural features than with those from noise features, supporting the intended separation of reward-relevant information.

D.2 Top-k Sensitivity and Agreement with Full Correction

Sensitivity to the number of corrected actions. We evaluate $k \in \{1, 2, 3, 4, 5\}$ on *3s5z_vs_3s6z* and *MMM2*, using three independent training seeds. Table 3 reports test win rates after 3M and 2M environment steps, respectively.

Table 3: Top-k sensitivity: test win rate (%), reported as mean $\pm$ standard deviation over three training seeds.

k	*3s5z_vs_3s6z* 3M steps	*MMM2* 2M steps
1	49.53 ± 6.15	88.28 ± 5.66
2	64.30 ± 3.35	96.25 ± 3.55
3	62.75 ± 7.07	92.97 ± 5.08
4	68.13 ± 5.47	90.47 ± 4.20
5	57.13 ± 7.12	93.13 ± 6.44

The highest mean win rate occurs at $k = 2$ on *MMM2* and $k = 4$ on *3s5z_vs_3s6z*. Among the tested settings, $k = 2$ has the smallest across-seed standard deviation on both maps and uses fewer candidate evaluations than larger values of k. These results support it as a practical fixed setting across the two tasks.

Action agreement with full correction. We compare action choices offline using four independently trained models per map and 64 fixed held-out episodes per training seed. The evaluation contains 113,104 valid state–agent decisions on *MMM2* and 91,952 on *3s5z*. For each decision, we use the same base utilities, residual predictions, and available-action mask to reconstruct the top-k and full-action correction rules. This comparison changes only the number of corrected actions, without retraining or changing the evaluated histories.

Let a_0, a_k, and a_{full} denote the actions selected by the base utilities, top-k correction, and full-action correction, respectively. We define

$$\begin{aligned} \text{SwitchRate}(k) &= \widehat{\Pr}(a_k \neq a_0), \\ \text{Recovery}(k) &= \widehat{\Pr}\left(a_k = a_{\text{full}} \mid a_{\text{full}} \neq a_0\right), \end{aligned} \tag{31}$$

where $\widehat{\Pr}$ denotes empirical frequency within each training seed's held-out decisions. SwitchRate measures how often top-k correction changes the base action. Recovery measures agreement with full correction specifically among decisions where full correction changes the base action. The full-action rule uses the same learned residual head and serves as a comparison of correction budgets, not as an optimal-action label.

Table 4: Offline comparison of action choices. All entries are percentages, reported as mean $\pm$ standard deviation across four training seeds.

	MMM2		*3s5z*	
k	SwitchRate (%)	Recovery (%)	SwitchRate (%)	Recovery (%)
1	0.5137 ± 0.4643	62.39 ± 9.42	0.2610 ± 0.0677	35.98 ± 8.04
2	0.2378 ± 0.1695	97.46 ± 4.05	0.1827 ± 0.0431	98.96 ± 1.47
3	0.2087 ± 0.1247	99.73 ± 0.53	0.1762 ± 0.0461	100.00 ± 0.00
4	0.1972 ± 0.1052	99.73 ± 0.53	0.1762 ± 0.0461	100.00 ± 0.00
5	0.1954 ± 0.1020	100.00 ± 0.00	0.1762 ± 0.0461	100.00 ± 0.00

With $k = 1$, correction changes the base action more often but agrees less frequently with the decisions changed by full correction. With $k = 2$, the corresponding agreement reaches 97.46% on

MMM2 and 98.96% on *3s5z*. Increasing k from 2 to 3 gives a smaller additional gain on this measure while increasing the candidate residual-evaluation budget by 50%. These results favor $k = 2$ as a low-budget approximation to the action changes made by full correction on the evaluated histories.

D.3 Encoder Comparisons and Computational Cost

We replace the Mamba structural encoder with a GRU or Transformer while keeping the rest of SND unchanged. Table 5 reports mean test win rates, parameter counts, FLOPs, and inference latency. The computational measurements use the same hardware and input configuration.

Table 5: Structural-encoder comparison within SND. Win-rate columns report means (%). Latency is measured under the same input configuration.

Encoder	*3s5z_vs_3s6z* 3M steps	*MMM2* 2M steps	Parameters	FLOPs	Latency (ms)
Mamba	64.30	96.25	821.91K	3.36G	52.89
GRU	35.78	92.89	787.09K	3.16G	55.44
Transformer	54.69	96.19	1.18M	6.37G	175.01

Mamba and Transformer achieve similar mean win rates on *MMM2*, while Mamba achieves a higher mean on *3s5z_vs_3s6z*. Mamba has slightly more parameters and FLOPs than GRU, but its measured latency is close to GRU and lower than Transformer under the same workload. These comparisons support Mamba as a practical choice for the structural encoder. The base ACORM learner takes 27.20 ms under the same profiling configuration, compared with 52.89 ms for SND, making the additional computational cost explicit.

D.4 Functional Specialization and Cross-Branch Dependence

We examine the information retained by the two branches on held-out *MMM2* trajectories, with both representation encoders frozen. Probe-training and test sets are separated at the episode level, and both branches are evaluated using the same standardized probe protocol. All results in this subsection are reported as mean $\pm$ standard deviation over five independently trained seeds.

Return and TD-residual predictability. To control for episode progress, we probe the timestep-residualized five-step discounted return

$$\widetilde{G}_t^{(5)} = G_t^{(5)} - \widehat{\mathbb{E}}_{\text{train}}\left[G_t^{(5)} \mid t\right], \tag{32}$$

where $G_t^{(5)}$ denotes the five-step discounted return and the conditional mean is estimated exclusively from probe-training episodes. For TD-residual prediction, each probe is conditioned on the corresponding branch representation and the recorded action. Table 6 reports test R^2 for both targets.

Table 6: Frozen-encoder probing on held-out *MMM2* episodes. Entries are test R^2, reported as mean $\pm$ standard deviation over five training seeds.

Prediction target	Structural z^s	Noise z^n
Timestep-controlled five-step return	0.10 ± 0.04	0.01 ± 0.02
Action-conditioned TD residual	0.16 ± 0.05	0.04 ± 0.03

The structural branch achieves higher test R^2 for both targets, indicating that return-related and correction-relevant information is more readily decoded from z^s under the evaluated probes. Removing $\mathcal{L}_{\text{adv}}$ increases the return-probe R^2 of z^n from 0.01 ± 0.02 to 0.08 ± 0.03 (Table 7), consistent with the adversarial objective suppressing return predictability in the noise branch.

Complementary behavioral information. We measure how much the noise logits improve action reconstruction beyond the structural logits. Let $\ell_s = W_s z^s$ and $\ell_n = W_n z^n$. The test-set cross-

entropy reduction is

$$\Delta_{\mathrm{CE}} = \widehat{\mathbb{E}}_{\mathrm{test}}\big[\,\mathrm{CE}(\mathrm{softmax}(\ell_s), a) - \mathrm{CE}(\mathrm{softmax}(\mathrm{sg}(\ell_s) + \ell_n), a)\big]. \tag{33}$$

A positive value means that adding the noise logits improves prediction of the recorded actions. The full model achieves $\Delta_{\mathrm{CE}} = 0.105 \pm 0.028$, compared with 0.014 ± 0.011 when $\mathcal{L}_n$ is removed. This result supports the role of residual behavior modeling in retaining complementary action information in z^n.

Table 7: Targeted objective ablations on *MMM2*. The first row measures return information decoded from z^n; the second measures the action-reconstruction improvement obtained by adding the noise logits. Entries are mean $\pm$ standard deviation over five training seeds.

Diagnostic	Full SND	Ablated objective	Ablation result
Return-probe $R^2(z^n)$	0.01 ± 0.02	w/o $\mathcal{L}_{\mathrm{adv}}$	0.08 ± 0.03
Residual action Δ_{CE}	0.105 ± 0.028	w/o $\mathcal{L}_n$	0.014 ± 0.011

Cross-branch dependence. We quantify dependence between the frozen branch representations using a shuffled-corrected Kraskov mutual-information estimate,

$$\widehat{I}_{\mathrm{excess}} = \widehat{I}_{\mathrm{KSG}}(Z^s; Z^n) - \mathbb{E}_\pi\left[\widehat{I}_{\mathrm{KSG}}(Z^s; \pi(Z^n))\right], \tag{34}$$

where π randomly permutes sample correspondence between the two branches. As a complementary measure of shared representational geometry, we report linear centered kernel alignment (CKA). For column-centered representation matrices X and Y with corresponding samples in their rows,

$$\mathrm{CKA}(X, Y) = \frac{\|X^\top Y\|_F^2}{\|X^\top X\|_F \, \|Y^\top Y\|_F}. \tag{35}$$

Here CKA is computed between the structural and noise representations of the same held-out samples.

Table 8: Cross-branch dependence on held-out *MMM2* trajectories. Entries are mean $\pm$ standard deviation over five training seeds.

Variant	Excess MI	Cross-branch CKA
Full SND	0.06 ± 0.02	0.14 ± 0.04
w/o $\mathcal{L}_{\mathrm{orth}}$	0.23 ± 0.05	0.31 ± 0.08

The full model has lower excess MI and cross-branch CKA than the variant without $\mathcal{L}_{\mathrm{orth}}$, indicating lower measured dependence and less shared representational geometry. Together, the probes, residual action-prediction diagnostic, and targeted objective ablations provide complementary evidence for the functional specialization of the two branches.

E Hyperparameter Configuration

Experiments are conducted on a dedicated server equipped with GPUs equivalent to NVIDIA RTX 4090 and 256GB of RAM. Our framework is implemented using PyTorch 1.13.1, CUDA 11.3, and Python 3.10. Evaluation is performed every 10,000 environment steps. The main benchmark results are averaged over five random seeds, while the additional analyses above report their corresponding seed counts separately. The key hyperparameters are provided below.

Table 9: Key hyperparameters used in our experiments.

Hyperparameter	**Value**
Orthogonality loss weight λ_o	0.002
Adversarial loss weight λ_a	0.1
TD residual loss weight λ_t	0.2
Total disentanglement loss weight λ_{dis}	0.01
Mamba hidden size d_{model}	128
Mamba state size d_{state}	16
Mamba convolution width d_{conv}	4
Mamba expansion factor	2
Top-k for TD correction	2
TD correction strength α	0.1
TD residual clipping threshold c	3.0
Batch size	32
Learning rate	6×10^{-4}