MA-JEPA: JOINT-EMBEDDING WORLD MODELS FOR MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models improve sample efficiency by training policies on imagined trajectories, but their usefulness depends on learning representations that capture the information needed for future control. We study whether self-supervised joint-embedding prediction (JEPA) can provide this learning signal in both single- and multi-agent reinforcement learning. We introduce MA-JEPA, a stochastic world model that replaces observation reconstruction with prediction of target representations, enabling model-based multi-agent reinforcement learning with centralized training and decentralized execution. A categorical latent state and a causal Transformer are trained with posterior, action-conditioned dynamics, and masked spatial prediction objectives and are then used for actor-critic learning from imagination in the latent space. For multi-agent control, we add a training-only joint predictor that conditions on joint local states and the actions to predict each agent's next local observation representation. These predictions are passed through the same local posterior used during real interaction with a centralized critic that is used only for value learning, with execution remaining decentralized. Our experiments show that this architecture performs strongly on SMAC, matching or exceeding the strongest reported comparator median on three of six evaluated maps.

1 INTRODUCTION

Model-based reinforcement learning (MBRL) learns an environment model and uses it to generate trajectories for policy improvement. By training an actor and critic on these imagined trajectories, methods can reuse each real transition for many learning updates (Hafner et al., 2019; 2020; 2025). The benefit, however, depends on the quality of the world model. Errors can accumulate as the model is rolled forward, and a policy trained inside the model may exploit predictions that do not match the real environment (Janner et al., 2019; Moerland et al., 2023).

Multi-agent MBRL adds a second problem, where an agent's transition probabilities generally depend on the actions of the other agents. For example, in the cooperative combat environments of the StarCraft Multi-Agent Challenge (SMAC), whether an opponent remains alive after one step depends on which target the whole team attacks. A local world model conditioned only on one agent's action cannot fully predict this outcome. A centralized model can condition on the joint action, but directly using a centralized latent state for policy learning leads to a different problem, where the actor is then trained on information that is unavailable during decentralized execution. Existing multi-agent model-based reinforcement learning does not fully address the problem (Egorov & Shpilman, 2022; Venugopal et al., 2024; Zhang et al., 2025; Deihim et al., 2025).

In this paper, we directly tackle this problem by introducing MA-JEPA, the first JEPA model-based multi-agent reinforcement learning (MARL) method. It achieves centralized training with decentralized execution (CTDE) by training on the same latent representation of observations in centralized training that it does in decentralized execution. (Oliehoek et al., 2008; Kraemer & Banerjee, 2016) Our agent explicitly maintains an independent recurrent state in a parameter-shared local world model. This enables us both to train agents using local latent states and to maintain parallel, independent copies of the world model during execution, each with information specific to an individual agent.

We train this model with a joint-embedding predictive architecture (JEPA) objective that predicts target observation embeddings from latent states and action histories, without reconstructing observations,

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.

unlike leading model-based MARL methods (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). During multi-agent training, a joint predictor conditions on all agents' local states and actions to predict an embedding of each agent's next local observation. The joint predictor does not construct the actor state directly. Instead, its prediction is passed through the same local posterior that processes an encoded real observation. During real interaction, this posterior combines the local dynamics state with the observed input's embedding. During centralized imagination, it combines the local dynamics state with the embedding predicted from the joint action. The actor, therefore, receives the same type of locally constructed state in both cases. A centralized critic uses all local states for value learning. Importantly, neither it nor the joint predictor runs during execution, meaning only agent-specific information is used during execution.

We first evaluate the local world model on the visual DeepMind Control (DMC) to test whether self-supervised embedding prediction supports stochastic actor-critic imagination. We then evaluate the full method on the SMAC.

Our contributions include:

1. A model-based architecture for CTDE that predicts each agent's next local observation embedding from all agents' local states and actions, then uses the same local posterior as during execution to construct the actor state.
2. To our knowledge, the first JEPA-based MARL approach, which trains local and joint dynamics by predicting target embeddings instead of reconstructing observations, enabling the same representation-learning approach in single-agent and multi-agent imagination.

2 PRELIMINARIES

2.1 MODEL-BASED MULTI-AGENT REINFORCEMENT LEARNING

We consider a cooperative decentralized partially observable Markov decision process (Dec-POMDP) with N agents. At timestep t, the environment is in state $x_t \in \mathcal{X}$, agent i receives a local observation $\mathbf{o}_t^i \in \mathcal{O}^i$, and selects an action $\mathbf{a}_t^i \in \mathcal{A}^i$. The individual actions form the joint action

$$\mathbf{a}_t^{1:N} = (\mathbf{a}_t^1, \dots, \mathbf{a}_t^N), \tag{1}$$

after which the environment transitions according to $p(x_{t+1} \mid x_t, \mathbf{a}_t^{1:N})$ and produces a reward r_{t+1}.

Because the environment state is not available during decentralized execution, agent i acts from its local interaction history

$$\tau_t^i = (\mathbf{o}_1^i, \mathbf{a}_1^i, \dots, \mathbf{o}_t^i), \qquad \mathbf{a}_t^i \sim \pi^i(\cdot \mid \tau_t^i). \tag{2}$$

Under CTDE, information from all agents may be used to learn models and value functions, while the executed policy remains conditioned only on information available in τ_t^i.

Throughout the paper, superscript i indexes agents and $\mathbf{x}_t^{1:N} = (\mathbf{x}_t^1, \dots, \mathbf{x}_t^N)$ denotes a synchronized collection of agent-indexed quantities. We use hats for quantities generated by a learned model and $\mathrm{sg}(\cdot)$ for stop-gradient.

Model-based reinforcement learning learns environment dynamics from collected experience and uses the learned model to generate trajectories for policy improvement (Moerland et al., 2023). In the multi-agent setting, imagined rollouts are generated from replay-inferred states while actions are sampled from the decentralized policies, and the model predicts the quantities required to advance the rollout, including rewards and continuation factors $\widehat{r}_{t+1}^i$ and $\widehat{c}_{t+1}^i$. We define $\widehat{c}_{t+1}^i$ as the discounted continuation factor, i.e., it incorporates both discounting and episode termination.

A critic may use centralized information during training. Let $V^i(\widehat{\mathbf{s}}_t^{1:N})$ denote the value estimate for agent i from the synchronized imagined states. Imagined λ-returns are

$$G_t^{\lambda,i} = \widehat{r}_{t+1}^i + \widehat{c}_{t+1}^i \left[(1-\lambda) V^i(\widehat{\mathbf{s}}_{t+1}^{1:N}) + \lambda G_{t+1}^{\lambda,i} \right], \tag{3}$$

where $\lambda \in [0, 1]$ controls the trade-off between value bootstrapping and longer imagined returns (Sutton, 1988; Hafner et al., 2025). The actor is optimized from imagined returns while centralized information remains restricted to training.

2.2 JOINT-EMBEDDING PREDICTIVE ARCHITECTURES

Joint-embedding predictive architectures (JEPAs) learn representations by predicting the embedding of a target from a related context instead of reconstructing the target itself (LeCun, 2022; Assran et al., 2023). Let x denote a context and y a prediction target. An online encoder E_ϕ, a target encoder $E_{\bar{\phi}}$, and a predictor P_θ produce

$$\mathbf{e}_x = E_\phi(x), \qquad \bar{\mathbf{e}}_y = \mathrm{sg}\big(E_{\bar{\phi}}(y)\big)\,, \qquad \widehat{\mathbf{e}}_y = P_\theta(\mathbf{e}_x), \tag{4}$$

with objective

$$\mathcal{L}_{\mathrm{JEPA}} = D(\widehat{\mathbf{e}}_y, \bar{\mathbf{e}}_y)\,, \tag{5}$$

where D measures discrepancy in representation space. The target encoder is typically updated as an exponential moving average (EMA) of the online encoder (Grill et al., 2020; Assran et al., 2023). Since predictive agreement can admit collapsed representations, practical JEPA methods additionally employ architectural asymmetry or representation regularization; we use SIGReg (Balestriero & LeCun, 2025).

For sequential decision making, prediction is additionally conditioned on the actions connecting the current context to a future observation. At horizon h, this takes the general form

$$\widehat{\mathbf{e}}_{t+h} = P_\theta(\mathbf{c}_t, \mathbf{a}_{t:t+h-1})\,, \qquad \bar{\mathbf{e}}_{t+h} = \mathrm{sg}\big(E_{\bar{\phi}}(\mathbf{o}_{t+h})\big)\,, \tag{6}$$

where $\mathbf{c}_t$ denotes the available predictive context. Action-conditioned joint-embedding models have been used for latent prediction, planning, and control (Assran et al., 2025; Maes et al., 2026).

For the remainder of the paper, $\mathbf{e}_t^i = E_\phi(\mathbf{o}_t^i)$ denotes the online observation embedding of agent i, $\bar{\mathbf{e}}_t^i$ its stopped EMA target, and $\widehat{\mathbf{e}}_t^i$ a predicted observation embedding. We reserve $\mathbf{z}_t^i$ for the stochastic latent state inferred by the local world model. In MA-JEPA the joint model predicts observation embeddings, which are subsequently processed by the local world-model posterior to construct the latent state used by the policy.

3 RELATED WORK

3.1 JOINT-EMBEDDING PREDICTIVE ARCHITECTURES

Joint-embedding predictive architectures learn representations by predicting target embeddings rather than reconstructing observations (LeCun, 2022). I-JEPA applies this principle to spatially masked image regions, demonstrating that predictive objectives can learn semantic visual representations without pixel-level reconstruction (Assran et al., 2023). V-JEPA extends the approach to video by predicting masked spatiotemporal representations, while V-JEPA 2 scales video pretraining and introduces an action-conditioned predictor for physical reasoning and planning (Bardes et al., 2024; Assran et al., 2025). These works establish joint-embedding prediction as an effective alternative to reconstructive generative modeling for learning structured representations of visual dynamics.

Recent work has brought this principle closer to reinforcement learning and world modeling. LeWorld-Model (LeWM) trains an action-conditioned JEPA end-to-end from pixels and uses SIGReg to stabilize the learned latent space, showing that future representation prediction can support efficient model-based control without an observation decoder (Maes et al., 2026). TD-JEPA instead develops policy-conditioned long-horizon latent prediction for zero-shot reinforcement learning, using temporal-difference structure to learn representations of future dynamics across policies (Bagatella et al., 2025). Existing applications have primarily focused on single-agent representation learning, planning, and control. MA-JEPA extends this architectural family to MARL, using the learned embedding space both as the predictive target of each local world model and as the interface through which a centralized joint model captures multi-agent interactions.

3.2 MODEL-BASED REINFORCEMENT LEARNING

Modern model-based reinforcement learning learns compact world models and uses their predictions for planning or policy optimization. PlaNet introduced stochastic latent dynamics for planning from high-dimensional observations, and the Dreamer family instead trains actor-critic policies directly on latent imagined trajectories (Hafner et al., 2019; 2020; 2025). More recent world models increasingly

use sequence-modeling architectures. IRIS represents visual observations with discrete tokens and models their dynamics autoregressively with a Transformer, while STORM combines stochastic latent states with Transformer dynamics for efficient imagination (Micheli et al., 2023; Zhang et al., 2023).

Extending world models to MARL additionally requires modeling interactions between agents while preserving decentralized policies. Existing methods have largely adapted established single-agent world-model architectures to this setting. MAMBA extends Dreamer-style latent imagination to cooperative MARL, using parameter-shared local world models together with inter-agent communication (Egorov & Shpilman, 2022). DMAWM similarly builds on DreamerV2, separating local agent modules from a shared environment model and aligning imagined latent-state distributions with those inferred from local observations (Xue et al., 2026). In contrast, MATWM develops a multi-agent extension of the STORM architecture, combining stochastic Transformer dynamics with teammate prediction, decentralized imagination, and a semi-centralized critic (Deihim et al., 2025). MARIE also adopts Transformer-based world modeling, coupling decentralized agent dynamics with a centralized Perceiver that aggregates information across agents (Zhang et al., 2025). MABL instead introduces global and local latent variables so that centralized information can improve world-model learning while policies remain decentralized (Venugopal et al., 2024).

MA-JEPA follows a different architectural lineage. Rather than adapting a Dreamer- or STORM-style world model to the multi-agent setting, we are the first paper, to our knowledge, to leverage joint-embedding predictive world modeling in a MARL setting. MA-JEPA combines joint-embedding predictive learning with centralized imagination and decentralized latent-state based policies for execution. Unlike MARIE, which trains policies on reconstructed observations, our agents act on latent states maintained by their local world models. These states summarize each agent's observation-action history, allowing its policy to use the temporal representations learned through dynamics prediction. Both local and joint models learn by predicting observation representations without reconstructing observations, removing the requirement that these representations also preserve the information needed for reconstruction.

During training, a separate joint model captures interactions using all agents' local states and joint actions. While DMAWM generates imagined latent states through a shared environment model and trains them to be aligned with locally inferred states, our joint model predicts each agent's next observation embedding. These embeddings pass through the same local posterior used for real observations. The actor's latent state is therefore constructed by the same local mechanism during imagination and execution, requiring no additional context from the centralized training for local execution and removing a potential architectural source of training–execution mismatch.

4 Theoretical Background

We now establish convergence results for MA-JEPA's world model under assumptions similar to Klindt et al. (2026). In Section E we establish alternative theoretical results on convergence without relying on these latent space assumptions for our method.

Action-conditioned Gaussian identification. Following the notation of Klindt et al. (2026), let $z, z' \in \mathbb{R}^n$ be consecutive physical states. Let $x = g(z)$ be the joint observation and $y = f(x) = h(z) \in \mathbb{R}^n$ the learned representation, where f and g are measurable. Assume $z \sim \mathcal{N}(0, I_n)$ and let A be a joint action in a finite set $\mathcal{A}$, independent of z. For each action $a \in \mathcal{A}$, let $B_a \in \mathbb{R}^{n\times n}$ and assume

$$z' \mid z, A = a \sim \mathcal{N}(B_a z, I_n - B_a B_a^\top), \qquad \|B_a\|_{\mathrm{op}} \le 1.$$

Define $\kappa_B = \lambda_{\min}(\mathbb{E}_A[B_A B_A^\top]) - \mathbb{E}_A\|B_A\|_{\mathrm{op}}^4$ and assume $\kappa_B > 0$. Let $J_\omega(y, a)$ predict the next representation. Consider minimizing the population loss $\mathcal{L} = \mathbb{E}\|h(z') - J_\omega(h(z), A)\|^2$ over encoder and predictor classes, subject to $\mathbb{E}h(z) = 0$ and $\mathrm{Cov}(h(z)) = I_n$. Assume these classes contain the encoder $h_0(z) = Q_0 z$ and predictor $J_{\omega_0}(y, a) = Q_0 B_a Q_0^\top y$ for some orthogonal matrix Q_0.

For each observed action a, define $\widehat{C}_{m,a}$ as the empirical mean of $y'y^\top$ over samples with action a. Define $\widehat{V}_{m,a}$ as the empirical mean of $(y' - \widehat{C}_{m,a}y)(y' - \widehat{C}_{m,a}y)^\top$ over those same samples. For each observed action, set $\widehat{P}_m(\cdot \mid y, a) = \mathcal{N}(\widehat{C}_{m,a}y, \widehat{V}_{m,a})$. For an action not yet observed, define $\widehat{P}_m(\cdot \mid y, a)$ to be any fixed Gaussian distribution.

Theorem 1 (Identification and transition consistency)**.** *The minimum of the population loss $\mathcal{L}$ over the admissible encoder–predictor pairs is $n - \operatorname{tr}\mathbb{E}_A[B_A B_A^\top]$. For every globally minimizing pair, there is an orthogonal matrix Q such that its encoder satisfies $h(z) = Qz$ almost surely. For any such encoder, let $y' = h(z')$. The conditional distribution of y' given $y = h(z)$ and $A = a$ admits the Gaussian version*

$$P(\cdot \mid y, a) = \mathcal{N}(QB_aQ^\top y, I_n - QB_aB_a^\top Q^\top).$$

Hold this globally minimizing encoder fixed and collect m fresh independent and identically distributed triples (y, A, y') from the joint distribution induced by the assumed model for (z, A, z') and this fixed encoder.

As $m \to \infty$, $\widehat{P}_m(\cdot \mid y, a)$ converges weakly to $P(\cdot \mid y, a)$ almost surely for each fixed $y \in \mathbb{R}^n$ and every action $a \in \mathcal{A}$ with $\Pr(A = a) > 0$. For each such action satisfying $\|B_a\|_{\mathrm{op}} < 1$, this convergence also holds in total variation.

See proof in Appendix F.

This theorem establishes that by using JEPA, under similar assumptions to Klindt et al. (2026), we are able to learn a world model whose learned distribution over next-period predicted latent states z' given current latent states z and all actions converges to the true distribution of the z', conditional on latent states today and actions. This means our world-model effectively recovers the "true" law of motion in the latent-space, and that further, this true latent world model is normally distributed and can be exactly characterized under the Klindt et al. (2026) assumptions. This justifies the approach of using JEPA and provides a theoretical justification to our method in the following section.

5 METHOD

We introduce MA-JEPA, a joint-embedding predictive model-based MARL algorithm for centralized training with decentralized execution. Each agent maintains a local latent world model and acts from its own inferred state. During training, a joint predictor models the effect of synchronized agent states and actions by predicting each agent's next observation embedding. These predictions are passed through the same local posterior used for real observations, enabling joint imagination without constructing a centralized actor state. A centralized critic is used for policy learning.

5.1 LOCAL PREDICTIVE WORLD MODEL

We denote the observation encoder by E_ϕ and the local latent dynamics by parameters θ, comprising a causal Transformer T_θ, a history-conditioned prior p_θ, and an observation-conditioned posterior q_θ. Agent i is represented by a latent-variable summarizing the action-observation history (AOH) into a deterministic history state $\mathbf{h}_t^i$ and a categorical stochastic latent state the current environment $\mathbf{z}_t^i$, with

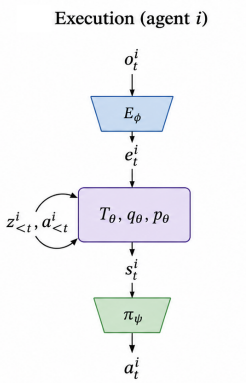
$$\mathbf{s}_t^i = \operatorname{cat}\big(\mathbf{h}_t^i, \operatorname{vec}(\mathbf{z}_t^i)\big) \tag{7}$$

forming the local state used by the policy.

Local state inference. The encoder represents the current observation, while the causal Transformer summarizes the preceding stochastic states and actions. The distribution over next state actions, which we call the *posterior* for simplicity combines both to infer the current stochastic state:

$$\begin{aligned} \mathbf{e}_t^i &= E_\phi(\mathbf{o}_t^i), \\ \mathbf{h}_t^i &= T_\theta(\mathbf{z}_{<t}^i, \mathbf{a}_{<t}^i), \\ \mathbf{z}_t^i &\sim q_\theta(\cdot \mid \mathbf{h}_t^i, \mathbf{e}_t^i). \end{aligned} \tag{8}$$

The prior $p_\theta(\mathbf{z}_t^i \mid \mathbf{h}_t^i)$ predicts the stochastic state from local history alone and is trained jointly with the posterior.



Decentralized execution. Agent i encodes its local observation and updates $\mathbf{s}_t^i$ from its own observation–action history. The shared policy then selects $\mathbf{a}_t^i \sim \pi_\psi(\cdot \mid \mathbf{s}_t^i)$ without access to other agents.

The deterministic state $\mathbf{h}_t^i$ summarizes information from the preceding latent–action history, whereas $\mathbf{z}_t^i$ incorporates information revealed by the current observation. This separation allows the local dynamics to advance without observing the next environment input while retaining an observation-conditioned correction when one becomes available. Parameters are shared across agents, while latent histories are maintained independently.

Learning joint-embeddings. We train two local predictors toward the EMA target of the embedding $\bar{\mathbf{e}}_t^i$ defined in Sec. 2.2. The posterior predictor P_{post} maps the complete local state $\mathbf{s}_t^i$ into the target embedding space, while the dynamics predictor P_{dyn} maps only the deterministic history state $\mathbf{h}_t^i$ into the same space:

$$\mathcal{L}_{\text{post}} = d_{\cos}\big(P_{\text{post}}(\mathbf{s}_t^i), \bar{\mathbf{e}}_t^i\big)\,, \qquad \mathcal{L}_{\text{dyn}} = d_{\cos}\big(P_{\text{dyn}}(\mathbf{h}_t^i), \bar{\mathbf{e}}_t^i\big)\,. \tag{9}$$

The two predictors impose complementary constraints on the local representation. P_{post} is evaluated after the current observation has been incorporated and therefore encourages $\mathbf{s}_t^i$ to retain the information represented by the target encoder. In contrast, P_{dyn} receives only $\mathbf{h}_t^i$ and must predict the same target from the preceding latent–action history. The resulting representation is therefore both descriptive of the current observation and predictable from local dynamics. Crucially, this embedding space later forms the interface through which the joint model supplies predicted observations to the local world model.

The complete local objective additionally contains prior–posterior latent regularization, reward and continuation prediction, and SIGReg. For visual observations, we also use masked spatial representation prediction.

Decentralized policy. Each agent acts only from its local latent state,

$$\mathbf{a}_t^i \sim \pi_\psi(\cdot \mid \mathbf{s}_t^i). \tag{10}$$

Policy parameters are shared across agents. The teammate-action augmentation used in our implementation depends only on $\mathbf{s}_t^i$ and is described in the appendix.

5.2 Joint Prediction of Multi-Agent Dynamics

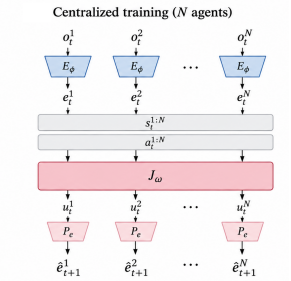
Local transition prediction does not account for the simultaneous actions of other agents. During centralized training, we therefore introduce a joint predictor J_ω conditioned on the synchronized local states $\mathbf{s}_t^{1:N}$ and joint action $\mathbf{a}_t^{1:N}$. It produces an agent-specific interaction representation,

$$\mathbf{u}_t^{1:N} = J_\omega\big((\mathbf{s}_t^{1:N}), \mathbf{a}_t^{1:N}\big)\,, \tag{11}$$

from which

$$\widehat{\mathbf{e}}_{t+1}^i = P_e(\mathbf{u}_t^i) \tag{12}$$

predicts the next local observation embedding of agent i. Conditioning on the joint action allows the predictor to model interaction effects that cannot be inferred from the focal agent's local history alone. The output remains agent-specific: the model predicts the representation of each agent's next local observation rather than constructing a shared actor state.



Centralized training. The joint model J_ω combines synchronized local states $\mathbf{s}_t^{1:N}$ with the joint action $\mathbf{a}_t^{1:N}$, forming agent-specific interaction representations $\mathbf{u}_t^i$ from which it predicts the next local embeddings $\widehat{\mathbf{e}}_{t+1}^i$.

Joint predictor training. The predicted embedding is supervised by both the EMA target and the stopped online encoding

$$\begin{aligned} \mathcal{L}_{\text{emb}} &= d_{\cos}\big(\widehat{\mathbf{e}}_{t+1}^i, \bar{\mathbf{e}}_{t+1}^i\big)\,, \\ \mathcal{L}_{\text{int}} &= \text{SmoothL1}\big(\widehat{\mathbf{e}}_{t+1}^i, \text{sg}(E_\phi(\mathbf{o}_{t+1}^i))\big)\,. \end{aligned} \tag{13}$$

The two targets serve different roles. The EMA target aligns joint prediction with the representation learned by the local JEPA objectives. The stopped online encoding matches the real encoding that the

local posterior receives during environment interaction. This objective reduces the mismatch between encoded observations and joint-model predictions before both are consumed by the same posterior during imagination.

Together with auxiliary transition predictions, the joint objective is

$$\mathcal{L}_{\text{joint}} = \beta_e \mathcal{L}_{\text{emb}} + \beta_I \mathcal{L}_{\text{int}} + \mathcal{L}^J_{\text{aux}}. \tag{14}$$

Here, $\mathcal{L}^J_{\text{aux}}$ includes reward, continuation, agent-alive, and legal-action prediction. In the method, factual reward, continuation, and agent-alive losses also update the local input representations at scale 1.0.

Multi-step prediction. We supervise future embeddings directly from replay of realized past states. Let $\mathcal{H}$ denote the set of prediction horizons and $\boldsymbol{\chi}^i_{t,h}$ the intervening action context used to predict the future of agent i. For each $h \in \mathcal{H}$, a head predicts

$$\widehat{\mathbf{e}}^i_{t+h|t} = P_h\big(\mathbf{u}^i_t, \boldsymbol{\chi}^i_{t,h}\big)\,, \qquad \mathcal{L}_{\text{MS}} = \sum_{h\in\mathcal{H}} \alpha_h\, d_{\cos}\Big(\widehat{\mathbf{e}}^i_{t+h|t}, \bar{\mathbf{e}}^i_{t+h}\Big)\,. \tag{15}$$

We train the predictor to predict future action sequences likely to be chosen by the individual agent. This action-sensitivity objective discourages future representations that explain the observed future independently of the actions that produced it. Together, direct prediction and action sensitivity encourage the joint representation to retain information that is both temporally predictive and relevant to control. The construction of $\boldsymbol{\chi}^i_{t,h}$ and the counterfactual objective are given in the appendix. These multi-step objectives are training auxiliaries and are not recursively used during imagination.

5.3 Joint Imagination and Decentralized Policy Learning

Shared posterior interface. During imagination, agents sample actions from their local policies and advance their local deterministic states. The joint predictor then produces the next observation embedding from the synchronized imagined states and joint action

$$\begin{aligned} \widehat{\mathbf{a}}^i_t &\sim \pi_\psi(\cdot \mid \widehat{\mathbf{s}}^i_t), \\ \widehat{\mathbf{h}}^i_{t+1} &= T_\theta(\widehat{\mathbf{z}}^i_{\le t}, \widehat{\mathbf{a}}^i_{\le t}), \\ \widehat{\mathbf{e}}^i_{t+1} &= P_e\big(J_\omega(\mathrm{sg}(\widehat{\mathbf{s}}^{1:N}_t), \widehat{\mathbf{a}}^{1:N}_t)_i\big)\,. \end{aligned} \tag{16}$$

The predicted embedding replaces the encoder output in the same posterior used for real observations

$$\begin{aligned} \mathbf{z}^i_{t+1} &\sim q_\theta(\cdot \mid \mathbf{h}^i_{t+1}, E_\phi(\mathbf{o}^i_{t+1})), \\ \widehat{\mathbf{z}}^i_{t+1} &\sim q_\theta(\cdot \mid \widehat{\mathbf{h}}^i_{t+1}, \widehat{\mathbf{e}}^i_{t+1}). \end{aligned} \tag{17}$$

The joint predictor therefore does not generate the next actor state directly. It predicts only the observation-side input to the local posterior. The posterior combines this prediction with the independently advanced local dynamics state in the same manner as it combines a real encoded observation with $\mathbf{h}^i_{t+1}$. Consequently, centralized information changes the predicted consequence of the joint action without changing the mechanism that constructs the decentralized policy state.

Centralized value Learning. The critic uses a central value network V_ξ that takes as inputs the stopped local states of all agents during training,

$$V^i_\xi = V_\xi\big(\mathrm{sg}(\widehat{\mathbf{s}}^{1:N}_t)\big)_i\,. \tag{18}$$

Using rewards and discounted continuations predicted by the joint model, we compute imagined λ-returns from Eq. equation 3. The centralized critic, therefore exploits full observability of the environment's current state for value estimation during training, while the actor continues to condition only on $\mathbf{s}^i_t$. The actor is optimized from the imagined returns and the critic from the corresponding value targets, with gradients stopped at the world-model states.

At execution time, each agent uses only the shared local encoder, latent dynamics, posterior, and policy. The joint predictor, direct multi-step objectives, and centralized critic are restricted to training.

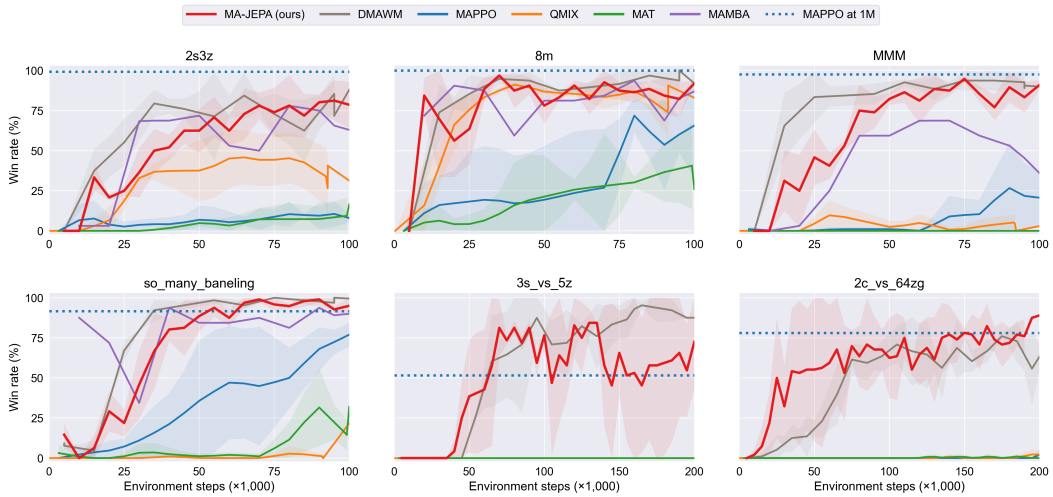


Figure 1: Win rate versus environment steps on six SMAC v1 tasks at the indicated budgets. Solid curves show means across available training seeds, with ± 1 sample-standard-deviation bands. Recorded evaluations are connected by linear interpolation without smoothing; The dotted horizontal line shows MAPPO's mean win rate at 1M steps.

6 Experiments

6.1 Experimental Setup

We evaluate whether joint-embedding world models support effective policy learning from latent imagination in the StarCraft Multi-Agent Challenge (SMAC; Samvelyan et al., 2019). Our suite comprises six SMAC v1 tasks under StarCraft II 4.10.0, covering homogeneous and heterogeneous teams, asymmetric engagements, and different coordination demands. A detailed description of SMAC can be found in Appendix B.1

We compare MA-JEPA against DMAWM, MAMBA, MAPPO, QMIX, and MAT at matched interaction budgets. MA-JEPA hyperparameters are given in Appendix C.9.1, and baseline configurations in Appendix C.9. MAPPO is additionally trained to 1M steps as a higher-budget reference. All algorithms were benchmarked under NVIDIA A100 PCIe GPUs.

6.2 Results

Table 1 reports win rates, and Figure 1 shows learning curves for the six tasks. These results demonstrate effective control through joint-embedding-based imagination.

Performance remains task-dependent. In particular, while we perform well on most tasks (winning on 3/6 and getting very close to DMAWM and near 100% win rate on `so_many_baneling`), our results on `3s_vs_4z` and the similar `3s_vs_5z` were noticeably weaker and had significantly higher variance. Agents were able to learn near optimal solutions on some of the seeds, but this was offset by poor performance on others. This is also true, though, to a lesser extent, for DMAWM. Our current hypothesis is that this high-variation in training results because these settings specifically study "kiting" behavior, where to win an agent must repeatedly approach, hit, and run away from the enemy. This is a multi-step sequential task that is highly recurrent throughout the episode and may be many steps beyond the rollout length of our current world model during imagination training. We therefore further hypothesize this sequence of actions is not currently captured effectively in the latent space of the world model consistently, leading the policy to fail to train.

In follow-up work, we plan to explore the geometry of the latent space embedding to understand exactly what is and is not learned by the world model in these "kiting" settings, what the optimal solution looks like, and identify optimal and learned policy features like recurrent action sequences.

Table 1: SMAC win rate (%) at matched environment-step budgets, reported as mean (sample standard deviation) across available training seeds from the set 1302, 2771, and 7636. Results use 100-episode evaluations at their final checkpoint. **Bold** denotes the highest available mean win rate for each map. Rows are grouped into easy and hard maps.

Map	Steps	MA-JEPA (ours)	DMAWM	MAPPO	QMIX	MAT
2s3z	100k	78.7 (14.5)	**88.0** (7.0)	8.0 (2.6)	31.3 (24.5)	16.5 (9.2)
8m	100k	**92.0** (1.0)	**92.0** (4.6)	65.7 (33.1)	83.0 (10.1)	26.0 (22.6)
MMM	100k	**91.0** (3.0)	90.0 (4.6)	20.7 (17.0)	3.0 (2.6)	0.0 (0.0)
so_many_baneling	100k	95.0 (6.1)	**99.5** (0.7)	77.0 (7.2)	21.7 (18.0)	32.0 (24.3)
3s_vs_5z	200k	72.5 (29.0)	**87.5** (17.7)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
2c_vs_64zg	200k	**80.3** (9.5)	75.3 (8.9)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)

6.3 Ablations

Figure 2 shows that the effects of the two gradient routes depend on the task. Removing factual JEPA gradients reduces performance on 2s3z but improves it on 3s_vs_4z. Conversely, removing outcome gradients improves performance on 2s3z, but yields lower mean performance and substantially greater seed variability on 3s_vs_4z. Enabling both routes provides a more balanced configuration across these tasks, although it does not achieve the highest mean on either task individually. We isolate how centralized prediction losses shape the representations used by decentralized policies. All four variants retain the local world model, joint predictor, outcome heads, and latent-imagination policy training. The ablation changes which factual joint losses additionally backpropagate into the local features supplied to the joint model.

We vary two gradient routes in a 2×2 factorial design. The *outcome route* propagates gradients from factual reward, continuation, and agent-alive prediction into the local input features. The *JEPA route* propagates the factual joint embedding-prediction error into those features at scale 0.1. This error compares the predicted embedding with a stopped exponential-moving-average target using cosine loss. The scale applies to the additional local-input gradient, not to the ordinary training loss of the joint predictor.

7 Conclusion and Future Work

We introduced MA-JEPA, a joint-embedding predictive world model for multi-agent reinforcement learning under centralized training and decentralized execution. Rather than reconstructing observations, the local world model learns predictive representations that support latent imagination and decentralized control. A training-only joint model captures interactions between agents by predicting each agent's next local observation embedding, which is processed by the same local posterior used during real interaction.

Our experiments show that this architecture performs strongly on SMAC at 50k environment steps, matching or exceeding the strongest reported comparator median on three of six evaluated maps.

These results suggest that joint-embedding prediction provides a viable alternative to reconstructive world models for both single- and multi-agent control. An important direction for future work is to understand which properties of the learned representations make them effective for world-model prediction and control. In particular, we are interested in whether predictive objectives recover representations that separate persistent physical dynamics, agent identity, and interaction-dependent information, and how such structure affects generalization across environments and teams. Extending MA-JEPA to broader continuous-control and heterogeneous multi-agent settings is another natural direction.

AI use statement

In this work, we used generative AI tools for assistance with coding, executing experiments, suggesting hyperparameters and experiments and architecture ablations, debugging, discovering some

of the related literature, further discussion about other literature, checking notational consistency and grammar, and the clarification and rewriting of paragraphs. We also used AI tools with repeated rounds of discussion and heavy guidance from the authors specifically in the construction of or expansion of some of the theorems.

In terms of required disclosures, we used AI at various points: to formulate (some) mathematical claims, (some) literature search, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods.

We have not used generative AI tools for idea generation, significant paper structure, interpret results, or theorem writing or proving without the human oversight and guidance described above. Therefore, tasks that include using AI to generate synthetic data sets, help develop theoretical models or conceptual frameworks, propose or refine hypotheses, assist with translation, clean and reformat datasets, and support qualitative and thematic data analysis are not applicable to this work

Additionally, we used generative AI tools to perform the following recommended disclosures to create or modify scientific figures or images, suggest experimental parameters, create or edit software code, creation of artifacts, draft parts of a research paper, sourcing/searching for information, identify relevant literature, edit a research paper to improve readability, and format references. We have reviewed all AI-assisted work to check its accuracy, both the theoretical work and its final generated content whenever sentences were rewritten or similar. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning, 2025.

Marco Bagatella, Matteo Pirotta, Ahmed Touati, Alessandro Lazaric, and Andrea Tirinzoni. TD-JEPA: Latent-predictive representations for zero-shot reinforcement learning, 2025.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.

Azad Deihim, Eduardo Alonso, and Dimitra Apostolopoulou. Transformer world model for sample efficient multi-agent reinforcement learning, 2025.

Vladimir Egorov and Aleksei Shpilman. Scalable multi-agent model-based reinforcement learning. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pp. 381–390, 2022.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

David Klindt, Yann LeCun, and Randall Balestriero. When does LeJEPA learn a world model?, 2026. URL https://arxiv.org/abs/2605.26379.

Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2016.01.031. URL https://www.sciencedirect.com/science/article/pii/S0925231216000783.

Yann LeCun. A path towards autonomous machine intelligence, 2022. URL https://openreview.net/forum?id=BZ5a1r-kVsf. Version 0.9.2.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023.

Thomas M. Moerland, Joost Broekens, Aske Plaat, and Catholijn M. Jonker. Model-based reinforcement learning: A survey. *Foundations and Trends in Machine Learning*, 16(1):1–118, 2023. doi: 10.1561/2200000086.

Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos Vlassis. Optimal and approximate q-value functions for decentralized pomdps. *J. Artif. Int. Res.*, 32(1):289–353, May 2008. ISSN 1076-9757.

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. PMLR, 2018. URL https://proceedings.mlr.press/v80/rashid18a.html.

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.

Henry Scheffé. A useful convergence theorem for probability distributions. *The Annals of Mathematical Statistics*, 18(3):434–438, 1947. doi: 10.1214/aoms/1177730390. URL https://doi.org/10.1214/aoms/1177730390.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3: 9–44, 1988.

Aravind Venugopal, Stephanie Milani, Fei Fang, and Balaraman Ravindran. MABL: Bi-level latent-variable world model for sample-efficient multi-agent reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pp. 1865–1873, 2024.

Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16509–16521, 2022. doi: 10.52202/068431-1201. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/69413f87e5a34897cd010ca698097d0a-Abstract-Conference.html.

Di Xue, Jing Jiang, Shaowei Zhang, Wenhao Guo, Lei Yuan, Zongzhang Zhang, and Yang Yu. Learning disentangled multi-agent world model for decentralized control. In *Forty-Third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nYyfpPubnW.

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, pp. 24611–24624, 2022. doi: 10.52202/068431-1787. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9c1535a02f0ce079433344e14d910597-Abstract.html.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. STORM: Efficient stochastic transformer based world models for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Yang Zhang, Chenjia Bai, Bin Zhao, Junchi Yan, Xiu Li, and Xuelong Li. Decentralized transformers with centralized aggregation are sample-efficient multi-agent world models. *Transactions on Machine Learning Research*, 2025.

A Comparing Multi-Agent World Model Architectures

Table 2: Comparison of MA-JEPA with multi-agent world model architectures.

Aspect	MARIE	MAMBA	MBVD	MATWM	DMAWM	MA-JEPA
Multi-Agent Design	Yes	Yes	Yes	Yes	Yes	Yes
Backbone Architecture	Transformer	GRU	GRU	Transformer	GRU + Transformer	Transformer
Latent Representation	VQ-VAE	Categorical VAE	VAE	Categorical VAE	Categorical RSSM	Categorical latent
Critic Type	Centralized	Centralized	Centralized	Semi-centralized	Centralized	Centralized
Agent Training	PPO-style	PPO-style	Deep Q-learning	DreamerV3-style	PPO-style	PPO-style

B Environment and Task Details

We evaluate MA-JEPA on SMAC (Samvelyan et al., 2019) Our task selection follows the evaluation settings used by closely related multi-agent world-model methods, enabling direct comparison under established benchmark configurations.

B.1 SMACv1

The StarCraft Multi-Agent Challenge (SMAC) consists of cooperative micromanagement scenarios in StarCraft II. Each allied unit is controlled by an individual agent and receives only a local observation of the battlefield. Observations contain information about visible allied and enemy units, including their relative positions, distances, health, shield values, and unit types. Agents select from discrete movement, stop, and unit-targeted attack actions. Invalid actions are excluded using the environment-provided legal-action mask.

SMAC is particularly suitable for evaluating multi-agent world models because an agent's local transition can depend strongly on the simultaneous actions of its teammates. For example, whether an enemy unit remains alive after an attack cannot in general be determined from the focal agent's action alone. MA-JEPA therefore uses the joint action during centralized world-model training while retaining a decentralized policy conditioned only on each agent's local interaction history.

We evaluate on the following six SMAC scenarios: `2s3z`, `8m`, `MMM`, `so_many_baneling`, `3s_vs_5z`, `2c_vs_64zg` These scenarios cover homogeneous and heterogeneous teams and

Table 3: SMACv1 scenarios used in our experiments.

Scenario	Difficulty	Training budget
2s3z	Easy	100K
8m	Easy	100K
MMM	Easy	100K
so_many_baneling	Easy	100K
3s_vs_5z	Hard	200K
2c_vs_64zg	Hard	200K

range from relatively simple coordination problems to difficult scenarios requiring precise joint behavior.

C Implementation Details

C.1 Replay Sampling

MA-JEPA maintains one replay buffer with two independently sampled views. The world-model view supplies model supervision, while the behavior view supplies the roots for imagined policy learning. Both views use the same mixture of uniform and exponential-recency sampling:

$$P(v) = \eta \frac{1}{|\mathcal{V}|} + (1-\eta)\frac{d^{\mathrm{age}(v)}}{\sum_{w\in\mathcal{V}} d^{\mathrm{age}(w)}}, \qquad \eta = 0.5, \quad d = 0.9998, \tag{19}$$

where $\mathcal{V}$ is the set of eligible sequence starts and age is measured by insertion order. The two views use independent random streams but may contain overlapping sequences. Independent sampling therefore separates the two batches without removing recency bias from the policy-root distribution.

Each replay sequence contains 192 context records followed by 64 learning records. The context records reconstruct the model's temporal history; no loss is computed on them.

For world-model training, we reconstruct local temporal states from stored latent–action inputs and stochastic states using the current temporal network. For policy and value training, performed after the world-model update, we re-encode the context observations and infer their posterior states using the updated model. In both cases, the resulting local features and recorded joint actions reconstruct the joint Transformer's temporal history.

Histories reset at episode boundaries. States supplied to policy and value learning are treated as fixed inputs, preventing these updates from changing the world model.

Sketched Isotropic Gaussian Regularization. To prevent representational collapse, we regularize the online observation embeddings with SIGReg (Balestriero & LeCun, 2025). For each agent slot, SIGReg samples random unit directions a_m and compares the empirical one-dimensional projected embedding distribution $u_{n,m} = a_m^\top e_n$ with a standard normal distribution using their characteristic functions. For projection m, the discrepancy is

$$\mathcal{T}_m = N \int \left| \frac{1}{N}\sum_{n=1}^{N} e^{\mathrm{i}tu_{n,m}} - e^{-t^2/2} \right|^2 e^{-t^2/2}\, dt, \tag{20}$$

and $\mathcal{L}_{\mathrm{SIGReg}}$ is obtained by averaging this statistic over random projections and nonempty agent slots. In practice, the integral is approximated on a finite frequency grid. By encouraging many random one-dimensional projections to resemble $\mathcal{N}(0,1)$, SIGReg promotes approximately isotropic, non-collapsed embeddings without requiring an observation reconstruction objective.

C.2 Local World-Model Objective

Let $e_t^i = E_\phi(o_t^i)$ denote the online observation embedding and $\bar{e}_t^i = \mathrm{sg}(E_{\bar{\phi}}(o_t^i))$ its stopped target, where $\bar{\phi}$ is an exponential moving average of ϕ. The local state is $s_t^i = [h_t^i; \mathrm{vec}(z_t^i)]$. Unless stated

otherwise, per-transition losses are averaged over their valid entries; invalid entries contribute zero, and an empty set of valid entries contributes zero. For the local objectives, validity is given by the replay presence and learning-activity masks, not necessarily physical controllability.

The two local JEPA objectives are

$$\mathcal{L}_{\text{post}} = \mathbb{E}_{\text{valid}}\big[d_{\cos}(P_{\text{post}}(s_t^i), \bar{e}_t^i)\big], \tag{21}$$

$$\mathcal{L}_{\text{dyn}} = \mathbb{E}_{\text{valid}}\big[d_{\cos}(P_{\text{dyn}}(h_t^i), \bar{e}_t^i)\big], \tag{22}$$

where $d_{\cos}(x, y) = 1 - \text{sim}_{\cos}(x, y)$. The posterior and prior are $q_t^i = q_\theta(\cdot \mid h_t^i, e_t^i)$ and $p_t^i = p_\theta(\cdot \mid h_t^i)$, respectively. Both use categorical probabilities with uniform mixing coefficient 0.01. The two directed regularizers are

$$\mathcal{L}_{\text{dyn}-\text{reg}} = \mathbb{E}_{\text{valid}}\left[\max\left\{\tau, D_{\text{KL}}\big(\text{sg}(q_t^i) \| p_t^i\big)\right\}\right], \tag{23}$$

$$\mathcal{L}_{\text{rep}-\text{reg}} = \mathbb{E}_{\text{valid}}\left[\max\left\{\tau, D_{\text{KL}}\big(q_t^i \| \,\text{sg}(p_t^i)\big)\right\}\right], \tag{24}$$

with free-nat threshold $\tau = 1$. KL divergences sum over the categorical variables before applying this threshold.

The complete local objective is

$$\begin{aligned}\mathcal{L}_{\text{local}} = {} & 2\mathcal{L}_{\text{post}} + 2\mathcal{L}_{\text{dyn}} + 0.05\mathcal{L}_{\text{SIGReg}} \\ & + \mathcal{L}_{\text{mask}}^{\text{local}} + \mathcal{L}_{\text{dyn}-\text{reg}} + 0.1\mathcal{L}_{\text{rep}-\text{reg}}.\end{aligned} \tag{25}$$

There are no local reward or continuation losses: these outcomes are predicted by the joint model. The vector-observation configuration contains neither an observation-reconstruction objective nor a spatial prediction objective.

Representation regularization. SIGReg is applied to online embeddings, grouping samples by agent slot across batch and time. For each nonempty group, it compares the empirical characteristic functions of 256 random unit-vector projections with the standard Gaussian characteristic function. The comparison uses 17 equally spaced points in $[0, 3]$ and Gaussian weighted quadrature. The projected discrepancy is multiplied by the group's valid sample count before averaging over nonempty groups. This sample-count scaling is part of the objective.

Action-availability supervision. The local availability head predicts a binary vector from s_t^i. For each vector, binary cross-entropy is averaged separately over legal and illegal actions, then averaged equally over the nonempty classes. The joint availability head uses the same balanced reduction.

C.3 One-Step Joint Prediction and Posterior Alignment

The joint model receives synchronized local states, the joint action, roster presence, and controllability. Agent self-attention first mixes information within a transition; a causal Transformer then maintains temporal context. Writing its cache explicitly as k_t, we obtain

$$(k_{t+1}, u_t^{1:N}) = J_\omega\big(k_t, \text{sg}(s_t^{1:N}), a_t^{1:N}, m_{\text{pres},t}^{1:N}, m_{\text{ctrl},t}^{1:N}\big), \tag{26}$$

$$\hat{e}_{t+1}^i = P_e(u_t^i). \tag{27}$$

No fixed agent-identity or agent-position embeddings are added. The detached local-state input specifies the baseline gradient boundary.

The embedding and interface objectives are

$$\mathcal{L}_{\text{emb}} = \mathbb{E}_{\text{valid}}\big[d_{\cos}(\hat{e}_{t+1}^i, \bar{e}_{t+1}^i)\big], \tag{28}$$

$$\mathcal{L}_{\text{int}} = \mathbb{E}_{\text{valid}}\big[\text{SmoothL1}(\hat{e}_{t+1}^i, \text{sg}(e_{t+1}^i))\big]. \tag{29}$$

Smooth-L_1 is averaged over embedding coordinates and uses a quadratic-to-linear transition at absolute error one. It penalizes pointwise discrepancy with the online encoder output; it does not establish equality of the real and imagined embedding distributions.

We additionally align the posterior distributions induced by real and predicted embeddings while holding the factual history fixed:

$$\mathcal{L}_{\text{align}} = \mathbb{E}_{\text{valid}}\Big[D_{\text{KL}}\big(\,\text{sg}(q_\theta(\cdot \mid h^i_{t+1}, e^i_{t+1})) \,\|\, q_{\theta^-}(\cdot \mid \text{sg}(h^i_{t+1}), \hat{e}^i_{t+1})\big)\Big], \tag{30}$$

where θ^- denotes a frozen copy of the current local parameters, not a separate learned posterior. The KL uses the same uniform-mixed categorical distributions as execution. Gradients pass through the predicted embedding to the joint model, but not to the factual history, target posterior, or local posterior parameters along this path.

The one-step joint objective is

$$\mathcal{L}_{\text{joint}} = 2\mathcal{L}_{\text{emb}} + \mathcal{L}_{\text{int}} + 0.05\mathcal{L}_{\text{align}} + \mathcal{L}^{\text{joint}}_r + \mathcal{L}^{\text{joint}}_c + \mathcal{L}^{\text{joint}}_{\text{mask}} + \mathcal{L}_{\text{alive}}. \tag{31}$$

Reward prediction uses a symlog two-hot categorical output. Continuation and controllability use binary outputs. With discounted continuation, the continuation target is $\gamma(1 - d^i_{t+1})$, where d denotes the replay terminal flag and $\gamma = 1 - 1/333$.

Embedding, interface, alignment, and next-mask losses require a controllable source, a present target, and no reset crossing. Death on the target transition remains a valid prediction event. Reward and continuation losses also supervise dead but present roster slots. Liveness supervision uses present source slots and excludes reset crossings. These masks are normalized separately.

C.4 Direct Multi-Step JEPA and Action Discrimination

Direct prediction heads are supervised at $\mathcal{H}_{\text{direct}} = \{1, 2, 4, 8\}$. The factual interaction feature u^i_t already contains the current joint action $a^{1:N}_t$. For $h > 1$, the corresponding head also receives the focal agent's recorded action tail:

$$\hat{e}^i_{t+h|t} = P_h(u^i_t, a^i_{t+1:t+h-1}). \tag{32}$$

The tail is empty for $h = 1$. Later teammate actions are not supplied by a teammate-plan model. These predictions are evaluated directly from factual roots; they are neither chained between horizons nor used to construct PPO states.

For ordered horizons $(h_0, h_1, h_2, h_3) = (1, 2, 4, 8)$, define

$$\alpha_{h_k} = \frac{\rho^k}{\sum_{j=0}^{3} \rho^j}, \qquad \rho = 0.75. \tag{33}$$

The direct loss is

$$\mathcal{L}_{\text{MS}} = \sum_{h \in \mathcal{H}_{\text{direct}}} \alpha_h \mathbb{E}_{\text{valid},h}\big[d_{\cos}(\hat{e}^i_{t+h|t}, \bar{e}^i_{t+h})\big]. \tag{34}$$

The available root indices leave room for the maximum horizon in the sampled suffix. Each horizon additionally requires a legal factual prefix, present and controllable action sources, and no reset crossing. Death exactly at the target is retained.

For each $h > 1$, action discrimination replaces the final focal-agent action a^i_{t+h-1} with every distinct alternative legal at that factual decision. The joint root and earlier focal actions remain unchanged. Each alternative incurs

$$\ell^i_{\text{AD},h} = \Big[m - \text{sim}_{\cos}(\hat{e}^i_{t+h|t}, \bar{e}^i_{t+h}) + \text{sim}_{\cos}(\hat{e}^{i,\text{alt}}_{t+h|t}, \bar{e}^i_{t+h})\Big]_+, \qquad m = 0.1. \tag{35}$$

We first average over legal alternatives at each root and then over valid roots. Horizon weights are renormalized over $\{2, 4, 8\}$ for the action loss. The resulting auxiliary is $2\mathcal{L}_{\text{MS}} + 0.1\mathcal{L}_{\text{AD}}$. The alternatives are discriminative negatives, not observed counterfactual targets; different legal actions need not always yield different observable outcomes.

C.5 Self-Fed Recurrent Supervision

In addition to direct prediction, we train recursive joint predictions under recorded joint actions. Starting from a factual root, the joint model predicts the next observation embedding, the local posterior constructs the next latent state, and that predicted state becomes the input to the following transition. Real future observations are used as targets, not injected into the predicted trajectory.

We sample up to eight eligible synchronized roots without replacement from the world-model batch. A root requires two valid subsequent transitions; later endpoint validity is assessed separately. The rollout extends for five transitions, with supervision at $\mathcal{H}_{\mathrm{SF}} = \{2, 4, 5\}$. Terminal arrivals are retained, while reset crossings, disappearing slots, and out-of-range endpoints are masked. Individual death does not itself define an episode boundary.

The joint model and reconstructed joint history use inference mode. Local transition parameters are frozen. Gradients through the joint temporal cache and the permitted local input paths are truncated by detaching recurrent state at the start of every two-step chunk. The joint predictor still detaches its local-state input at each step; this is not unrestricted end-to-end backpropagation through the full five-step simulator. The trajectory-alignment term retains the input Jacobian through the frozen local transition within each chunk.

Unlike one-step alignment, trajectory alignment compares factual and recursively generated histories:

$$\mathcal{L}_{\mathrm{traj},h} = \mathbb{E}_{\mathrm{valid},h}\Big[D_{\mathrm{KL}}\big(\,\mathrm{sg}(q_\theta(\cdot \mid h^i_{t+h}, e^i_{t+h})) \,\|\, q_{\theta^-}(\cdot \mid \hat{h}^i_{t+h}, \hat{e}^i_{t+h})\big)\Big]. \tag{36}$$

The total recurrent auxiliary is

$$\mathcal{L}_{\mathrm{SF}} = \frac{0.1}{|\mathcal{H}_{\mathrm{SF}}|} \sum_{h \in \mathcal{H}_{\mathrm{SF}}} \Big(2\mathcal{L}_{\mathrm{emb},h} + \mathcal{L}_{\mathrm{int},h} + \mathcal{L}^{\mathrm{joint}}_{r,h} + \mathcal{L}^{\mathrm{joint}}_{c,h} + \mathcal{L}^{\mathrm{joint}}_{\mathrm{mask},h} + \mathcal{L}_{\mathrm{alive},h} + 0.1\mathcal{L}_{\mathrm{traj},h}\Big). \tag{37}$$

Each endpoint loss is normalized over its valid roots and agents; endpoints receive equal weight. The trajectory-KL coefficient is therefore 0.01 relative to the endpoint-averaged trajectory loss. Shared reward and continuation losses include dead but present slots; the other endpoint losses require a controllable root.

This supervised rollout uses the joint availability head with thresholded masks. Recorded actions continue to drive the rollout even when excluded by its predicted mask. This differs from the local-head, Bernoulli-sampled masks used for PPO imagination below.

C.6 Joint Imagination and Decentralized Execution

After each world-model update, we reconstruct the independent behavior batch with the updated model. Its 64 suffix positions supply candidate imagination roots; episode-final roots are excluded by validity masks. Each valid root produces $H = 5$ imagined transitions.

At every transition, each agent samples an action from the shared actor using only its own local state and current action mask. The local history advances using that agent's latent and action. In parallel, the joint predictor consumes the synchronized states, joint action, and joint temporal cache to predict the next local embedding. The next stochastic state is then sampled as

$$\hat{z}^i_{t+1} \sim q_\theta(\cdot \mid \hat{h}^i_{t+1}, \hat{e}^i_{t+1}). \tag{38}$$

The local prior remains a training regularizer, but it is not the sampling distribution used for these joint imagined transitions.

Imagined action availability. Root masks come from replay. At subsequent states, the local availability head predicts probabilities from the completed local state. Each action's availability is sampled as a Bernoulli variable. An empty predicted mask falls back to the no-op action, and non-controllable agents receive no-op-only support. The effective masks actually used to sample actions are stored with the imagined batch and are not resampled during PPO updates.

SMAC roster and team outcomes. The fixed roster is retained after individual deaths. Imagined controllability is monotonic: an agent remains controllable only if it was controllable previously and the predicted next controllability probability is at least 0.5. Joint reward and continuation predictions are averaged over present slots and broadcast back to those slots. The all-dead team is absorbing: the transition into this condition retains its final reward, while continuation and later rewards are zero. These are the shared-reward, fixed-roster rules of the SMAC adapter, not assumptions about all benchmark environments.

Execution. Real interaction uses the observation encoder, local history model, local posterior, and actor, together with the environment-provided legal-action mask. Parameters are shared, but each agent maintains its own history. Neither the joint predictor nor the centralized critic is used during execution. There is no teammate-belief network or residual belief adapter in this configuration.

C.7 PPO AND REPLAY-VALUE LEARNING

One imagined batch is generated before actor–critic optimization. Its local features, centralized critic context, actions, effective masks, old policy logits, advantages, returns, and trajectory weights remain fixed throughout the update. The world model receives no PPO gradients.

Let $\bar{V}_t^i$ be the target critic prediction and let b_t^i indicate value-valid states. Define the effective discounted continuation $\tilde{c}_{t+1}^i = \hat{c}_{t+1}^i b_{t+1}^i$. Generalized advantages and return targets are

$$\delta_t^i = \hat{r}_{t+1}^i + \tilde{c}_{t+1}^i \bar{V}_{t+1}^i - \bar{V}_t^i, \tag{39}$$

$$A_t^i = b_t^i\big(\delta_t^i + \lambda \tilde{c}_{t+1}^i A_{t+1}^i\big), \qquad A_H^i = 0, \tag{40}$$

$$R_t^i = A_t^i + \bar{V}_t^i, \qquad \lambda = 0.95. \tag{41}$$

The continuation already includes discounting; it is not multiplied by γ a second time. Critic-valid states include dead but present SMAC agents, whereas actor-valid decisions additionally require controllability. An individual death therefore removes subsequent actor decisions without discarding the earlier shared-team return.

Losses use cumulative continuation weights, $w_t^i = b_t^i \prod_{j=1}^{t} \tilde{c}_j^i$, with an empty product of one. Actor advantages are normalized once using the weighted mean and variance over valid decisions in the frozen batch. Writing the normalized advantage as $\widetilde{A}_t^i$ and the stored mask as M_t^i, the probability ratio is

$$r_t^i(\psi) = \frac{\pi_\psi(a_t^i \mid s_t^i, M_t^i)}{\pi_{\text{old}}(a_t^i \mid s_t^i, M_t^i)}. \tag{42}$$

For numerical stability, its log is clipped to $[-20, 20]$ before exponentiation. The actor minimizes

$$\begin{aligned}\mathcal{L}_{\text{actor}} = -\mathbb{E}_{\text{actor},w}\Big[&\min\big(r_t^i \widetilde{A}_t^i, \text{clip}(r_t^i, 1-\epsilon, 1+\epsilon)\widetilde{A}_t^i\big) \\ &+ \beta_{\text{ent}} \mathcal{H}\big(\pi_\psi(\cdot \mid s_t^i, M_t^i)\big)\Big],\end{aligned} \tag{43}$$

where $\mathbb{E}_{\text{actor},w}$ is the normalized weighted mean over actor-valid decisions, $\epsilon = 0.2$, and $\beta_{\text{ent}} = 0.003$. Entropy is not divided by the size of the action support.

The centralized critic predicts a symlog two-hot value distribution and is trained with that distribution's target loss, rather than a clipped mean-squared-error value objective:

$$\mathcal{L}_{\text{critic}} = \mathbb{E}_{\text{value},w}\big[\ell_{\text{twohot}}(V_\xi, \text{sg}(R))\big] + 0.3\mathcal{L}_{\text{replay}-\text{value}}. \tag{44}$$

The replay-value term uses real rewards and current imagined root-return bootstraps to construct λ-returns with $\lambda_{\text{replay}} = 0.95$. It trains only the critic; recorded replay actions are not PPO actor samples. Terminal arrivals retain their rewards and stop bootstrapping, and traces never cross resets. When the adapter distinguishes nonterminal truncation, the final real observation supplies its bootstrap. The supplied SMAC adapter uses the same upstream termination flag for episode-final and terminal markers, so that distinct truncation case is not exercised by this adapter.

Five actor steps and five critic steps reuse the same full imagined batch, with actor and critic updates interleaved. No new trajectories or action masks are sampled between these steps. The target critic is copied from the updated critic after each active learner call; the additional slow-critic regularizer is disabled.

C.8 Optimizer Ownership and Update Order

Parameters are partitioned into four disjoint groups: the local world model, joint world model, actor, and centralized critic. The model objective is

$$\mathcal{L}_{\mathrm{WM}} = \mathcal{L}_{\mathrm{local}} + \mathcal{L}_{\mathrm{joint}} + 2\mathcal{L}_{\mathrm{MS}} + 0.1\mathcal{L}_{\mathrm{AD}} + \mathcal{L}_{\mathrm{SF}}. \tag{45}$$

The local group contains the encoder, local dynamics and posterior, local JEPA predictors, and local availability head. The joint group contains the interaction and temporal predictor, joint output heads, and direct multi-step heads. In the baseline, joint losses do not update the local-world parameters. Frozen local modules can still transmit gradients with respect to their inputs on the alignment and self-fed paths described above.

A learner call first differentiates $\mathcal{L}_{\mathrm{WM}}$ and updates the two model groups, leaving actor and critic optimizer states untouched. It then reconstructs the behavior view with the updated model, generates and freezes an imagined batch, and performs the separate PPO actor and critic steps. This is not a single combined world-model–actor–critic backward pass.

All groups use adaptive gradient clipping followed by RMS preconditioning and momentum. The local and joint world-model learning rates are 10^{-4}; actor and critic learning rates are 3×10^{-5}. The clipping coefficient is 0.3, momentum coefficients are $(0.9, 0.999)$, optimizer ϵ is 10^{-20}, and weight decay and learning-rate warm-up are disabled in this configuration. The target encoder is updated after a finite model update using

$$\bar{\phi} \leftarrow 0.99\bar{\phi} + 0.01\phi. \tag{46}$$

C.9 Baseline Hyperparameters

We benchmark DMAWM, MARIE, MAMBA, MAPPO, QMIX, and MAT using implementations released by their respective authors. The tables below report the effective settings used by our benchmark launchers, verified against the released implementations and saved run configurations.

Unless stated otherwise, all baselines use StarCraft II 4.10.0, AI difficulty 7, and `step_mul`=8. Training budgets count real environment transitions across collectors; evaluation interactions, imagined transitions, and individual-agent actions do not contribute to this count.

C.9.1 MA-JEPA HYPERPARAMETERS

We use a shared configuration across all SMAC tasks, changing only the task, number of agents, training seed, and interaction budget.

Table 4: MA-JEPA hyperparameters used in our SMAC experiments.

Hyperparameter	**Value used**
World model	
Observation encoder layers / width	3 / 1024
Deterministic state dimension	4096
Categorical latent	32×64
Latent uniform mixture	0.01
Local Transformer layers / width / heads	2 / 512 / 8
Local / joint context length	64 / 16
Joint agent-interaction / temporal layers	2 / 12
Joint Transformer width / heads	256 / 4
Joint dropout	0.1
World-model learning rate	10^{-4}
Adaptive gradient clipping	0.3
Target-encoder EMA rate	0.01
SIGReg coefficient	0.05
Direct JEPA prediction horizons	$\{1, 2, 4, 8\}$
Action-margin / loss coefficient	0.1 / 0.1
Self-fed horizons / BPTT length	$\{2, 4, 5\}$ / 2
Outcome / factual JEPA input-gradient scales	1.0 / 0.1
Policy learning	
Actor hidden layers / width	3 / 512
Central critic layers / width	2 / 256
Actor / critic learning rate	3×10^{-5} / 3×10^{-5}
Actor / critic PPO epochs	5 / 5
PPO clipping	0.2
Entropy coefficient	0.003
Discount / return-trace λ	$1 - 1/333$ / 0.95
Imagination horizon	5
Auxiliary replay-value loss coefficient	0.3
Imagined action mask	Local; Bernoulli sampling

C.9.2 DMAWM

DMAWM follows Xue et al. (2026) and the authors' released implementation.[1]

Table 5: DMAWM hyperparameters used in our benchmark.

Hyperparameter	**Value used**
Reinforcement learning	
Optimizer	Adam
Actor / critic learning rate	3×10^{-5} / 3×10^{-5}
Actor / critic gradient clipping	100 / 100
Entropy coefficient	0.01
PPO epochs / clipping	5 / 0.2
Discount γ / GAE λ	0.99 / 0.95
Replay-value loss coefficient	1.0
Importance-weight cap	4
Actor / critic sharing	True / True
World model	
World-model learning rate	10^{-4}
World-model gradient clipping	**1000**
Batch size / sequence length	16 / 64
Replay burn-in / imagination horizon	4 / 4
Replay capacity / prefill	250,000 / 5000
Training ratio	128
Dynamics / representation KL scales	1.0 / 0.1
KL free bits	1.0
Deterministic latent width	512
Categorical latent	32×32
Latent uniform mixture	0.01
Transition / global Transformer layers	3 / 3
Transformer heads	8
Observation decoder hidden width / layers	1024 / 2
Collection environments	16
Actor / critic MLP width / layers	256 / 2
Critic Transformer layers / heads	1 / 8

[1] https://github.com/DiXue98/DMAWM/tree/2acaaeb82805b55d275e3ce08ac8f713ec9afbb2

C.9.3 MAPPO AND MAT

MAPPO uses the official implementation of Yu et al. (2022).[2] MAT uses the official implementation of Wen et al. (2022).[3]

Table 6: MAPPO and MAT hyperparameters used in our benchmark.

Hyperparameter	**MAPPO**	**MAT**
Policy	Recurrent or FF	Transformer
Optimizer	Adam	Adam
Actor / critic learning rate	5×10^{-4} each	Joint 5×10^{-4}
Rollout length per collector	400	100
Parallel collection environments	8	32
Transitions per rollout batch	3200	3200
Entropy coefficient	0.01	0.01
Discount γ	0.99	0.99
GAE λ	0.95	0.95
PPO epochs	15 or 5	15
PPO clipping	0.2 or 0.05	0.05
Minibatches per PPO epoch	1	1
Maximum gradient norm	10	10
MLP / recurrent hidden width	64	—
Recurrent layers	1	—
Transformer blocks / heads	—	1 / 1
Transformer embedding width	—	64
Value normalization / PopArt	True / False	True / False
Policy / value active masks	True / False	True / False

[2] https://github.com/marlbenchmark/on-policy/tree/de66d7a4b23fac2513f56f96f73b3f5cb96695ac

[3] https://github.com/PKU-MARL/Multi-Agent-Transformer/tree/be3ff49c8264d454c1fe2c41582aa2bfc98498c8

C.9.4 QMIX

We evaluate QMIX (Rashid et al., 2018) using the optimized PyMARL2 implementation.[4] The settings combine PyMARL2's QMIX, default, and SMAC configurations.

Table 7: QMIX/PyMARL2 hyperparameters used in our benchmark.

Hyperparameter	**Value used**
Agent architecture / hidden width	Recurrent / 64
Optimizer / learning rate	Adam / 10^{-3}
Discount γ / TD λ	0.99 / 0.6
Target-network update interval	200 episodes
Maximum gradient norm	10
Training batch size	128 episodes
Replay capacity	5000 episodes
Parallel collection environments	8
Exploration ϵ	$1.0 \rightarrow 0.05$
Linear epsilon annealing duration	100K environment steps
Mixer embedding / hypernetwork width	32 / 64
Agent-ID / previous-action inputs	True / True
Layer normalization	False
Orthogonal initialization	False
Prioritized replay	False

C.9.5 MAMBA

MAMBA follows Egorov & Shpilman (2022) and the authors' released StarCraft. [5]

Table 8: MAMBA hyperparameters used in our benchmark.

Hyperparameter	**Value used**
World-model / actor / critic optimizer	Adam / Adam / Adam
World-model learning rate	2×10^{-4}
Actor / critic learning rate	5×10^{-4} / 5×10^{-4}
Actor weight decay	10^{-5}
World-model / policy gradient clipping	100 / 100
Replay capacity / minimum replay	250,000 / 500
World-model / policy batch size	40 / 40
Replay sequence length	20
Imagination horizon	15
World-model optimizer steps per cycle	60
Policy updates per cycle	4
PPO epochs / clipping	5 / 0.2
Discount γ / return λ	0.99 / 0.95
Initial entropy coefficient	0.001
Entropy multiplier per actor step	0.99998
Deterministic latent width	256
Categorical latent	32×32
Actor / critic hidden width / layers	256 / 2
Collection environments	1
Collected-sample gate	1; updates after complete episodes

D ADDITIONAL ABLATIONS

[4] https://github.com/hijkzzz/pymarl2/tree/8ccac7c5aa134422a2e3009be735d23cdf8ce2f8

[5] https://github.com/jbr-ai-labs/mamba/tree/2c97258f71bf1c421c40ce14fd2f7cc3fe7fe19f

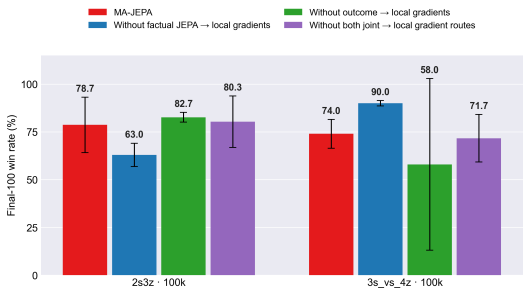


Figure 2: Ablation of joint-to-local gradient routes on two SMAC tasks at 100k environment steps. MA-JEPA enables outcome gradients at scale 1.0 and factual JEPA gradients at scale 0.1 into local representations. Each variant disables one or both routes while retaining the joint prediction objectives. Bars show mean win rate from completed 100-episode final evaluations; error bars indicate sample standard deviation across training seeds.

E Consistency of Learned Joint Successor Law with Fixed Representations

Consistency of the joint successor law. Fix MA-JEPA's parameters and collection policy. Consider independent and identically distributed episodes of $T < \infty$ steps. Let $\mathbf{z}_t$ denote the joint latent state, $\mathbf{a}_t \in \mathcal{A}$ the joint action, and $\mathbf{m}_t$ the joint legal-action mask. For a fixed history-window length $w \geq 1$, define the padded context $\mathbf{c}_t = (t, \mathbf{z}_{t-w+1:t}, \mathbf{a}_{t-w:t-1}, \mathbf{m}_{t-w+1:t}) \in \mathcal{C}$ and the complete successor $\mathbf{y}_{t+1} = (\mathbf{z}_{t+1}, \tilde{r}_{t+1}, e_{t+1}, \mathbf{m}_{t+1}) \in \mathcal{Y}$. Here $\tilde{r}$ is a quantized reward and $e \in \{0, 1\}$ indicates termination. All alphabets are finite. Let $P(\cdot \mid \mathbf{c}, \mathbf{a})$ denote the conditional distribution of the complete successor under the fixed parameters and collection policy. We suppress the conditioning arguments in the bounds below.

Theorem 2 (Joint-law consistency)**.** *From n sampled episodes, let $n(\mathbf{c}, \mathbf{a})$ count occurrences of context–action pair $(\mathbf{c}, \mathbf{a})$. At each observed pair, let $\widehat{P}_n(\cdot \mid \mathbf{c}, \mathbf{a})$ be the empirical distribution of its complete successors. At an unobserved pair, define $\widehat{P}_n$ to be any fixed probability distribution on $\mathcal{Y}$.*

As $n \to \infty$, $\widehat{P}_n$ converges to P almost surely in total variation, uniformly over all context–action pairs with positive probability of occurring under the fixed collection policy. For fixed n, $n_0 \geq 1$, and $0 < \delta < 1$, the following bound holds with probability at least $1 - \delta$, simultaneously for all pairs with $n(\mathbf{c}, \mathbf{a}) \geq n_0$.

$$\mathrm{TV}(\widehat{P}_n, P) \leq \min\left\{1, \sqrt{\frac{|\mathcal{Y}| \log 2 + \log(2|\mathcal{C}||\mathcal{A}|/\delta)}{2n_0}}\right\}.$$

Conditional on observing a given pair exactly $s \geq 1$ times, its expected total-variation error satisfies

$$\mathbb{E}\big[\mathrm{TV}(\widehat{P}_n, P) \mid n(\mathbf{c}, \mathbf{a}) = s\big] \leq \tfrac{1}{2}\sqrt{(|\mathcal{Y}| - 1)/s}.$$

At each context–action pair with positive occurrence probability, consider an unrestricted autoregressive model of the complete successor $\mathbf{y}$. Its population teacher-forced cross-entropy is minimized uniquely by the joint distribution $P(\cdot \mid \mathbf{c}, \mathbf{a})$.

See proof in Appendix F.

F Proofs of Theoretical Results

This extends the Hermite argument of Klindt et al. (2026) to joint-action prediction. The full covariance allows cross-agent dependence. The result recovers population optima of a modified objective followed by estimation with a fixed encoder.

Proof of Theorem 1. Identification. Decompose the centered encoder into its linear and higher Hermite degrees:

$$h(z) = Mz + \nu(z), \qquad M = \mathbb{E}[h(z)z^\top], \qquad W = \mathbb{E}\,\|\nu(z)\|^2.$$

Whitening gives $MM^\top + \mathbb{E}[\nu\nu^\top] = I_n$, hence $I_n - M^\top M \succeq 0$ and $\operatorname{tr}(I_n - M^\top M) = W$.

Let $(T_a v)(z) = \mathbb{E}[v(z') \mid z, A = a]$. Under the singular value decomposition $B_a = U \operatorname{diag}(\sigma_j) V^\top$, the rotated coordinates $V^\top z, U^\top z'$ form independent Gaussian pairs with correlations σ_j. The Hermite generating function maps normalized products of coordinate degrees α_j between these rotated bases with multipliers $\prod_j \sigma_j^{\alpha_j}$. These products form orthonormal Gaussian L_2 bases, and rotations preserve total degree and norm. Since ν contains only degrees at least two, $T_a(Mz) = MB_a z$ is orthogonal to $T_a\nu$ and $\|T_a\nu\|_{L_2}^2 \le \|B_a\|_{\mathrm{op}}^4 W$. Write $\Sigma_B = \mathbb{E}_A[B_A B_A^\top]$, $\beta_B = \mathbb{E}_A\|B_A\|_{\mathrm{op}}^4$, and $\mathcal{L}^\star = n - \operatorname{tr}\Sigma_B$. Conditioning on (z, A) gives

$$\begin{aligned}\mathcal{L} &\ge n - \mathbb{E}_A \|T_A h\|_{L_2}^2 \ge n - \operatorname{tr}(M\Sigma_B M^\top) - \beta_B W \\ &= \mathcal{L}^\star + \operatorname{tr}\{\Sigma_B(I_n - M^\top M)\} - \beta_B W \ge \mathcal{L}^\star + \kappa_B W.\end{aligned}$$

The feasible pair (h_0, J_{ω_0}) has residual $Q_0(z' - B_A z)$, whose expected squared norm is $\mathcal{L}^\star$. Thus every optimum has $W = 0$; whitening makes $M = Q$ orthogonal. Substitution gives the stated Gaussian conditional law.

Transition estimation. For each action with positive probability, its sample count diverges almost surely. Let $S_{m,a}, R_{m,a}$ average $yy^\top, y'y'^\top$ within action a. The strong law gives $\widehat{C}_{m,a} \to C_a := QB_aQ^\top$ and $S_{m,a}, R_{m,a} \to I_n$ almost surely. Expanding the fitted residual moment,

$$\widehat{V}_{m,a} = R_{m,a} - 2\widehat{C}_{m,a}\widehat{C}_{m,a}^\top + \widehat{C}_{m,a} S_{m,a} \widehat{C}_{m,a}^\top \longrightarrow I_n - C_a C_a^\top.$$

Gaussian parameter convergence gives weak convergence for each fixed y. If $\|B_a\|_{\mathrm{op}} < 1$, the limiting covariance is positive definite, as are the estimated covariances eventually. Their densities converge pointwise, so Scheffé's lemma (Scheffé, 1947) gives total-variation convergence. □

Theorem 2 (Joint-law consistency)**.** *From n sampled episodes, let $n(\mathbf{c}, \mathbf{a})$ count occurrences of context–action pair $(\mathbf{c}, \mathbf{a})$. At each observed pair, let $\widehat{P}_n(\cdot \mid \mathbf{c}, \mathbf{a})$ be the empirical distribution of its complete successors. At an unobserved pair, define $\widehat{P}_n$ to be any fixed probability distribution on $\mathcal{Y}$.*

As $n \to \infty$, $\widehat{P}_n$ converges to P almost surely in total variation, uniformly over all context–action pairs with positive probability of occurring under the fixed collection policy. For fixed n, $n_0 \ge 1$, and $0 < \delta < 1$, the following bound holds with probability at least $1 - \delta$, simultaneously for all pairs with $n(\mathbf{c}, \mathbf{a}) \ge n_0$.

$$\mathrm{TV}(\widehat{P}_n, P) \le \min\Big\{1, \sqrt{\tfrac{|\mathcal{Y}| \log 2 + \log(2|\mathcal{C}||\mathcal{A}|/\delta)}{2n_0}}\Big\}.$$

Conditional on observing a given pair exactly $s \ge 1$ times, its expected total-variation error satisfies

$$\mathbb{E}\big[\mathrm{TV}(\widehat{P}_n, P) \mid n(\mathbf{c}, \mathbf{a}) = s\big] \le \tfrac{1}{2}\sqrt{(|\mathcal{Y}| - 1)/s}.$$

At each context–action pair with positive occurrence probability, consider an unrestricted autoregressive model of the complete successor $\mathbf{y}$. Its population teacher-forced cross-entropy is minimized uniquely by the joint distribution $P(\cdot \mid \mathbf{c}, \mathbf{a})$.

Proof of Theorem 2. Condition on the episodes containing a fixed positive-probability pair $(\mathbf{c}, \mathbf{a})$. Our setting ensures one outcome per selected episode, so these outcomes are iid from P. Hoeffding's inequality over the $2^{|\mathcal{Y}|}$ outcome subsets, averaged over counts $s \ge n_0$, gives, for $\eta > 0$,

$$\Pr\big(\mathrm{TV}(\widehat{P}_n, P) > \eta,\ n(\mathbf{c}, \mathbf{a}) \ge n_0\big) \le 2^{|\mathcal{Y}|+1} e^{-2n_0\eta^2}.$$

A union bound over pairs gives the probability bound. For $p_{\mathbf{y}} = P(\mathbf{y} \mid \mathbf{c}, \mathbf{a})$, binomial variance and Cauchy–Schwarz give

$$\mathbb{E}[\mathrm{TV}(\widehat{P}_n, P) \mid s] \le \frac{1}{2\sqrt{s}} \sum_{\mathbf{y}} \sqrt{p_{\mathbf{y}}(1 - p_{\mathbf{y}})} \le \frac{1}{2}\sqrt{\frac{|\mathcal{Y}| - 1}{s}}.$$

For positive-probability pairs, the strong law gives convergence of the empirical count ratios. Finitely many pairs and outcomes then give uniform almost-sure convergence.

For the autoregressive claim, fix such a pair and order the coordinates of the complete outcome $\mathbf{y}$. Every joint law admits a factorization $\widetilde{q}(\mathbf{y}) = \prod_j \widetilde{q}_j(y_j \mid y_{<j})$. Its teacher-forced cross-entropy decomposes into entropy and Kullback–Leibler divergence:

$$-\mathbb{E}_P \sum_j \log \widetilde{q}_j(y_j \mid y_{<j}) = H(P) + D_{\mathrm{KL}}(P\|\widetilde{q}),$$

so its unique minimizing joint law is P. □