Revisiting Multi-Agent World Modeling from a Diffusion-Inspired Perspective

Yang Zhang[1]*, **Xinran Li[2], Jianing Ye[3], Delin Qu[4], Shuang Qiu[5],**
Chongjie Zhang[3], Xiu Li[1], Chenjia Bai[6]†

[1]Tsinghua University, [2]The Hong Kong University of Science and Technology,
[3]Washington University in St. Louis, [4]Fudan University, [5]City University of Hong Kong,
[6]Institute of Artificial Intelligence (TeleAI), China Telecom
`z-yang21@mails.tsinghua.edu.cn, baicj@chinatelecom.cn`

Abstract

World models have recently attracted growing interest in Multi-Agent Reinforcement Learning (MARL) due to their ability to improve sample efficiency for policy learning. However, accurately modeling environments in MARL is challenging due to the exponentially large joint action space and highly uncertain dynamics inherent in multi-agent systems. To address this, we reduce modeling complexity by shifting from jointly modeling the entire state-action transition dynamics to focusing on the state space alone at each timestep through sequential agent modeling. Specifically, our approach enables the model to progressively resolve uncertainty while capturing the structured dependencies among agents, providing a more accurate representation of how agents influence the state. Interestingly, this sequential revelation of agents' actions in a multi-agent system aligns with the reverse process in diffusion models—a class of powerful generative models known for their expressiveness and training stability compared to autoregressive or latent variable models. Leveraging this insight, we develop a flexible and robust world model for MARL using diffusion models. Our method, **D**iffusion-**I**nspired **M**ulti-**A**gent world model (DIMA), achieves state-of-the-art performance across multiple multi-agent control benchmarks, significantly outperforming prior world models in terms of final return and sample efficiency, including MAMuJoCo and Bi-DexHands. DIMA establishes a new paradigm for constructing multi-agent world models, advancing the frontier of MARL research.

1 Introduction

Learning accurate world models to capture environmental dynamics is crucial for effective decision-making. In the realm of model-based reinforcement learning (MBRL), such models play a pivotal role by enabling policy training through learning in imagination [1, 2, 3, 4], facilitating planning with look-ahead search [5, 6], or combining both approaches [7, 8]. While MBRL has achieved significant success in single-agent settings, extending these methodologies to multi-agent scenarios presents unique challenges, necessitating new approaches for multi-agent world modeling.

In multi-agent settings, where multiple agents simultaneously interact within a shared environment, two primary challenges emerge. First, the joint action space grows exponentially with the number of agents [9, 10], making it computationally expensive to directly handle joint dynamics. Second, the complex interdependencies among agents [11] make it difficult to accurately capture how individual

---

*Work done during the internship at TeleAI.
†Corresponding Author.

Preprint. Under review.

actions impact global state transitions. Current multi-agent world modeling approaches face a fundamental tradeoff. On one end of the spectrum, centralized modeling schemes directly capture full joint dynamics but incur computational costs that scale exponentially with the number of agents. On the other end, decentralized approaches [12, 13, 14] model individual agent dynamics separately and rely on additional mechanisms, such as sophisticated communication or aggregation modules, to recover the global state. However, this misalignment between decentralized model structure and the global Markov decision process (MDP) can impose inherent limitations on model accuracy and those communication or aggregation modules do not have explicit signal for supervision, further hindering the training. This tradeoff motivates a fundamental rethinking of the world model structure: Can we develop a centralized modeling scheme that maintains global consistency without auxiliary components in decentralized methods, while keeping computational complexity manageable as the number of agents increases?

To address this challenge, we adopt a sequential agent modeling perspective that processes agents' actions incrementally, as illustrated in Figure 1. Specifically, consider a multi-agent system at timestep t with global state s_t. When all agents' actions $a_t^{1:n}$ are unknown, the next state s_{t+1} remains highly uncertain. As agents' actions are progressively revealed, this uncertainty gradually decreases. This sequential uncertainty reduction process bears striking similarity to the reverse process in diffusion models [15, 16, 17], where generation is framed as iterative denoising from noise to clean samples.

Inspired by this conceptual similarity and the recent success of diffusion models in image-based world modeling [18, 19, 20], we propose **D**iffusion-**I**nspired **M**ulti-**A**gent world model (DIMA), which reformulates multi-agent dynamics prediction as a modified conditional denoising process. Despite employing a centralized modeling scheme, DIMA achieves computational complexity that scales linearly with the state space dimensionality, regardless of the number of agents. We summarize our contributions as follows:
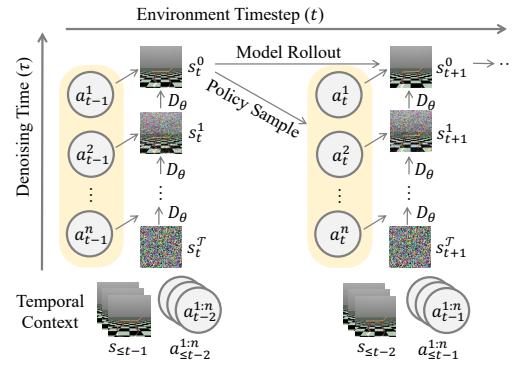

Figure 1: Illustration of the DIMA world model. From the temporal perspective, each environmental timestep is modeled as a complete denoising process, analogous to diffusion models. Within each timestep, we further consider an agent-wise perspective, where the introduction of each individual agent's action information represents a single denoising step, progressively reducing uncertainty about the next state.

- We leverage the connection between sequential agent modeling and diffusion processes to reformulate multi-agent dynamics prediction as a conditional denoising process. This enables a centralized modeling scheme that reduces complexity without additional communication mechanisms.

- We propose DIMA, a centralized multi-agent world model tailored for model-based MARL, and derive its corresponding evidence lower bound (ELBO), providing theoretical insights. We then instantiate DIMA within the EDM training framework [21] and integrate it into the learning-in-imagination paradigm for policy optimization.

- We evaluate DIMA on challenging continuous MARL benchmarks, including MAMuJoCo [22] and Bi-DexHands [23], in low-data regimes. Experimental results show that DIMA consistently improves the prediction accuracy of environment dynamics and outperforms both model-free and strong model-based MARL baselines in terms of sample efficiency and overall performance.

2 Preliminaries

2.1 Multi-Agent Systems as Dec-POMDP

We focus on fully cooperative multi-agent systems where all agents share a team reward signal. We formulate the system as a decentralized partially observable Markov decision process (Dec-POMDP) [11], which can be described by a tuple $(\mathcal{N}, \mathcal{S}, \mathcal{A}, P, R, \mathbf{\Omega}, \mathcal{O}, \gamma)$. $\mathcal{N} = \{1, ..., n\}$ denotes a set of agents, $\mathcal{S}$ is the finite global state space, $\mathcal{A} = \prod_{i=1}^{n} \mathcal{A}^i$ is the product of finite action spaces of all agents, i.e., the joint action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the global transition probability

function, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the shared reward function, $\boldsymbol{\Omega} = \prod_{i=1}^{n} \Omega^i$ is the product of finite observation spaces of all agents, i.e., the joint observation space, $\boldsymbol{\mathcal{O}} = \{\mathcal{O}^i, i \in \mathcal{N}\}$ is the set of observing functions of all agents. $\mathcal{O}^i : \mathcal{S} \to \Omega^i$ maps global states to the observations for agent i, and γ is the discount factor. Given a global state s_t at timestep t, agent i is restricted to obtaining solely its local observation $o_t^i = \mathcal{O}^i(s_t)$, takes an action a_t^i drawn from its policy $\pi^i(\cdot|o_{\leq t}^i)$ based on the history of its local observations $o_{\leq t}^i$, which together with other agents' actions gives a joint action $\boldsymbol{a}_t = (a_t^1, ..., a_t^n) \in \boldsymbol{\mathcal{A}}$, equivalently drawn from a joint policy $\boldsymbol{\pi}(\cdot|\boldsymbol{o}_{\leq t}) = \prod_{i=1}^{n} \pi^i(\cdot|o_{\leq t}^i)$. Then the agents receive a shared reward $r_t = R(s_t, \boldsymbol{a}_t)$, and the environment moves to next state s_{t+1} with probability $P(s_{t+1}|s_t, \boldsymbol{a}_t)$. The aim of all agents is to learn a joint policy $\boldsymbol{\pi}$ that maximizes the expected discounted return $J(\boldsymbol{\pi}) = \mathbb{E}_{s_0, \boldsymbol{a}_0, ... \sim \boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \boldsymbol{a}_t) \right]$. Note that recent approaches [12, 13, 14] build the multi-agent world models via modeling $P(\boldsymbol{o}_{t+1}|\boldsymbol{o}_t, \boldsymbol{a}_t)$ in the joint observation and action space, which mismatches with the transition formulation in Dec-POMDP. However, DIMA is trained to recover the well-defined global state transition $P(s_{t+1}|s_t, \boldsymbol{a}_t)$ according to the proposed multi-agent dynamics formulation.

2.2 Score-based Diffusion Models

In this work, we directly utilize the unified framework and the accompanying practical design choice of diffusion models introduced by Karras et al. [21].

Notation. Let us consider a diffusion process $\{\mathbf{x}^\tau\}_{\tau \in [0, \mathcal{T}]}$ indexed by a continuous time variable $\tau \in [0, \mathcal{T}]$, with corresponding marginals $\{p^\tau\}_{\tau \in [0, \mathcal{T}]}$, and boundary conditions $p^0 = p_{\text{data}}$ and $p^{\mathcal{T}} = p_{\text{prior}}$, where p_{prior} is usually a pure Gaussian distribution in practical implementation. For clarity, we use the superscript τ to denote the diffusion process timestep and the subscript t to denote the trajectory timestep.

ODE Expression. Song et al. [17] models the forward and reverse diffusion processes with stochastic differential equations (SDEs) which describe how the desired distribution of sample $\mathbf{x}$ evolves over time τ. Assuming the stochasticity only comes from the initial sample $\mathbf{x}^{\mathcal{T}}$ of prior distribution p_{prior}, Karras et al. [21] expresses diffusion models via its corresponding probability flow ordinary differential equation (ODE) [17] which continuously increases or reduces the noise level of the image when moving forward or backward in time, respectively. The defining characteristic of the probability flow ODE is that evolving a sample $\mathbf{x}^{\tau_a} \sim p^{\tau_a}(\mathbf{x}) = p(\mathbf{x}; \sigma(\tau_a))$ from time τ_a to τ_b (either forward or backward in time) yields a sample $\mathbf{x}^{\tau_b} \sim p^{\tau_b}(\mathbf{x}) = p(\mathbf{x}; \sigma(\tau_b))$, where $\sigma(\tau)$ is a schedule that defines the desired noise level at time τ. It is described by

$$d\mathbf{x} = -\dot{\sigma}(\tau)\sigma(\tau)\nabla_{\mathbf{x}} \log p^\tau(\mathbf{x}) \, d\tau,$$

where the dot denotes a time derivative. $\nabla_{\mathbf{x}} \log p^\tau(\mathbf{x})$ is the score function [24] associated with the marginals $\{p^\tau\}_{\tau \in [0, \mathcal{T}]}$ along the process. Equipped with the score function, we can thus smoothly mold random noise into data for sample generation, or diffuse a data point into random noise.

Denoising Score Matching. By using the score matching objective [24], we can evaluate the score function easily. Specifically, $D_\theta(\mathbf{x}; \tau)$ is a parameterized denoiser function that minimizes the expected L_2 denoising error for samples $\mathbf{x}^0$ drawn from p_{data} for every $\sigma(\tau)$,

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}^0 \sim p_{\text{data}}(\mathbf{x}), \mathbf{x}^\tau \sim p(\mathbf{x}^\tau | \mathbf{x}^0)}[\|D_\theta(\mathbf{x}^\tau; \tau) - \mathbf{x}^0\|^2], \tag{1}$$

where $\mathbf{x}^\tau \sim p(\mathbf{x}^\tau|\mathbf{x}^0)$ denotes that $\mathbf{x}^\tau$ is obtained by applying Gaussian noise of scale $\sigma(\tau)$ to clean sample $\mathbf{x}^0$. Then the score estimation can be given by $\nabla_{\mathbf{x}} \log p^\tau(\mathbf{x}) = (D_\theta(\mathbf{x}; \tau) - \mathbf{x})/\sigma(\tau)^2$ at any given time τ. Thanks to this estimation, we can solve the ODE by numerical integration, i.e., taking finite steps over discrete time intervals with the help of various ODE solvers.

3 Methodology

In the following, we first elucidate our proposed formulation of modeling multi-agent dynamics from a diffusion-inspired perspective in §3.1. Based on this formulation, we derive the corresponding ELBO and score matching objective for implementing the diffusion model that incorporates such a perspective. Then, we describe the behavior learning process within the world model in §3.2.

3.1 Modeling Multi-Agent Dynamics from a Diffusion-Inspired Perspective

Given a dataset $\{(\boldsymbol{o}_1, s_1, \boldsymbol{a}_1, r_1, \ldots, \boldsymbol{o}_{T_i}, s_{T_i}, \boldsymbol{a}_{T_i}, r_{T_i})\}_i$ containing all collected episodes, the aim of the multi-agent world model is to precisely predict how the next state is like based on an action intervention, i.e., recovering the unknown ground truth environment dynamics $P(s_{t+1}|s_t, \boldsymbol{a}_t)$.

Diffusion-Inspired Formulation. Supposing there are n agents $\{1, 2, \ldots, n\}$ and n noise levels $\{\sigma_n, \ldots, \sigma_2, \sigma_1\}$ that satisfy $\sigma_{\max} = \sigma_n > \cdots > \sigma_1 > \sigma_0 = 0$, the noisy sample $s_{t+1}^{(i)}$ is corrupted from the clean next state $s_{t+1}^{(0)} := s_{t+1}$ by adding noise of the corresponding level σ_i. Note that here we use the superscript to denote the diffusion process timestep except for the action notation a. Following the definition in [25], we start by defining a similar conditional Markovian forward diffusion process $\hat{q}$,

$$\hat{q}(s_{t+1}^{(0)}) := p(s_{t+1}), \tag{2}$$

$$\hat{q}(s_{t+1}^{(k+1)}|s_{t+1}^{(k)}, s_t, a_t^{1:n}) := q(s_{t+1}^{(k+1)}|s_{t+1}^{(k)}), \tag{3}$$

$$\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1}^{(0)}, s_t, a_t^{1:n}) := \prod_{k=1}^{n} \hat{q}(s_{t+1}^{(k+1)}|s_{t+1}^{(k)}, s_t, a_t^{1:n}), \tag{4}$$

where q denotes the unconditional forward diffusion process. While the conditional forward diffusion process $\hat{q}$ is conditioned on the control signal $(s_t, a_t^{1:n})$, we can prove that it behaves exactly like the unconditional one q. The following equations hold,

$$\hat{q}(s_{t+1}^{(k+1)}|s_{t+1}^{(k)}) = \hat{q}(s_{t+1}^{(k+1)}|s_{t+1}^{(k)}, s_t, a_t^{1:n}), \; \hat{q}(s_{t+1}^{(1):(n)}|s_{t+1}^{(0)}) = q(s_{t+1}^{(1):(n)}|s_{t+1}^{(0)}). \tag{5}$$

The detailed proof is referred to Dhariwal and Nichol [25]. Since the above equations suggest that the forward diffusion process is independent of the control signal $(s_t, a_t^{1:n})$, we can now fully focus on describing our formulation via the conditional reverse diffusion process.

To describe how the predicted next state gets sharpened progressively with sequentially given action of each agent, we have to specify the conditioning order. Without loss of generality, we adopt the descending order of agent id $(n, n-1, \ldots, 1)$ as the conditioning order. Formally, we make the following assumption in terms of the global state transition.

Assumption 1 (Diffusion-Inspired Decomposition of Multi-Agent Dynamics). *In our diffusion-inspired formulation with the descending order of agent id $(n, n-1, \ldots, 1)$ as the conditioning order, the global state transition $P(s_{t+1}|s_t, a_t^{1:n})$ yields the next state in a manner akin to a typical reverse diffusion process, i.e., satisfying*

$$P(s_{t+1}, s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}) = p(s_{t+1}^{(n)}) \prod_{k=1}^{n} p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, a_t^k, s_t), \tag{6}$$

where $s_{t+1}^{(n)}$ is corrupted with the noise of maximum level σ_n, practically indistinguishable from pure Gaussian noise.

Under the assumption, we have the following new form of Evidence Lower Bound (ELBO) on the $\log P(s_{t+1}|s_t, a_t^{1:n})$.

Theorem 2 (ELBO under the Diffusion-Inspired Formulation). *Under Assumption* 1*, the log-likelihood of the multi-agent global state transition (i.e., the evidence of the transition) is lower bounded as follows,*

$$\log P(s_{t+1}|s_t, a_t^{1:n}) \geq \underbrace{\mathbb{E}_{q(s_{t+1}^{(1)}|s_{t+1}^{(0)})}[\log p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, a_t^1, s_t)]}_{\text{reconstruction term}} - \underbrace{D_{\text{KL}}(q(s_{t+1}^{(n)}|s_{t+1}^{(0)})\|p(s_{t+1}^{(n)}))}_{\text{prior matching term}}$$

$$- \sum_{k=2}^{n} \underbrace{\mathbb{E}_{q(s_{t+1}^{(k)}|s_{t+1}^{(0)})}\left[D_{\text{KL}}(q(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})\|p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, a_t^k, s_t))\right]}_{\text{denoising matching term}}. \tag{7}$$

The detailed proof is deferred to §A. The *denoising matching term* in Eq. (7) secretly reveals that we can learn a parameterized denoising intermediate step $p_\theta(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, a_t^k, s_t)$ that matches

the tractable ground-truth denoising intermediate step $q(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})$, thereby realizing the formulation we propose. When we utilize Gaussian noise for corruption in the forward diffusion process, the *denoising matching term* can be simplified as a variant of Eq. (1):

$$\mathcal{L}(\theta) = \mathbb{E}\left[\sum_{k=1}^{n} \|D_\theta(s_{t+1}^{(k)}; \sigma_k, s_t, a_t^k) - s_{t+1}\|^2\right], \text{ given the order } (n, \ldots, 2, 1) \qquad (8)$$

However, there are still two properties to be incorporated into Eq. (8). **(i) Permutation Invariance.** Note that our formulation merely provides a novel perspective for modeling multi-agent dynamics, rather than changing the underlying mechanism of global state transitions. In other words, regardless of how the conditioning order of $a_t^{1:n}$ is specified, the next state should remain unchanged given the same current state and joint action, i.e., exhibiting *permutation invariance*. Therefore, for any possible order $\rho = (i_1, i_2, \ldots, i_n)$ uniformly sampled from the whole permutation set $\text{Perm}\{1, 2, \ldots, n\}$, we should optimize an expectation of Eq. (8) over the whole permutation set. **(ii) Condition-Independent Noising Process.** According to Eqs. (2)-(5), the conditional forward diffusion process is independent of the conditions. It allows us to randomly sample the noise levels $\{\sigma_1, \ldots, \sigma_n\}$ with the predefined continuous-time noise scheduler $\sigma(\tau)$ in §2.2.

Putting the above two together, we finally derive the optimization objective of DIMA,

$$\mathcal{L}(\theta) = \mathbb{E}_{\{\sigma_1,\ldots,\sigma_n\}\sim\sigma(\tau)}\mathbb{E}_{\rho\sim\text{Perm}\{1,2,\ldots,n\}}\left[\sum_{k=1}^{n}\|D_\theta(s_{t+1}^{(k)}; \sigma_k, s_t, a_t^{i_k}) - s_{t+1}\|^2\right]$$

$$= \mathbb{E}_\tau\mathbb{E}_{k\sim\text{Uniform}\{1,2,\ldots,n\}}\left[\|D_\theta(s_{t+1}^\tau; \sigma(\tau), s_t, a_t^k) - s_{t+1}\|^2\right], \qquad (9)$$

where $k \sim \text{Uniform}\{1, 2, \ldots, n\}$ indicates that the agent index k is uniformly sampled from the set $\{1, 2, \ldots, n\}$.

Comparison with Conventional Approaches. We present a concise illustration to highlight the fundamental difference between our DIMA and recent diffusion-based methods [26, 27] in modeling multi-agent dynamics. As shown in Figure 2, recent methods attempt to inject the entire joint action information into the progressively denoised next state at every intermediate step, whereas DIMA incorporates only a single agent's action at each step. Denoting the state space size as $|\mathcal{S}|$ and the individual action space size as $|\mathcal{A}|$, DIMA compresses the relevant information from a $|\mathcal{S}| \times |\mathcal{A}| \times |\mathcal{S}|$ space into a $|\mathcal{S}|$ space for each intermediate state transition $p_\theta(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, a_t^k, s_t)$. In contrast, existing methods must handle a significantly higher cost due to compressing information from a

Figure 2: Comparison between conventional flattened multi-agent modeling and DIMA's sequential agent modeling. Light gray indicates clean states; dark gray indicates noisy states.

$|\mathcal{S}| \times |\mathcal{A}|^n \times |\mathcal{S}|$ space into $|\mathcal{S}|$. This simple qualitative analysis demonstrates that despite modeling multi-agent dynamics in a centralized manner, DIMA enjoys a linear complexity in modeling difficulty with respect to the number of agents.

Practical Implementation. Inspired by the success of DIAMOND [18], a powerful single-agent diffusion-based world model, we adopt a similar design choice and employ the EDM framework [21] to effectively train the desired diffusion model. Specifically, the denoiser D_θ is reparameterized using the EDM preconditioners as follows:

$$D_\theta(s_{t+1}^\tau; \sigma(\tau), s_t, a_t^k) = c_{\text{skip}}^\tau s_{t+1}^\tau + c_{\text{out}}^\tau F_\theta(c_{\text{in}}^\tau s_{t+1}^\tau; c_{\text{noise}}^\tau, s_t, a_t^k), \qquad (10)$$

where F_θ is the neural network. These preconditioners $(c_{\text{skip}}^\tau, c_{\text{out}}^\tau, c_{\text{in}}^\tau, c_{\text{noise}}^\tau)$ are detailed in §B. In addition, we incorporate two practical techniques to further improve DIMA's predictive performance: (i) we maintain a running mean and standard deviation of global states to normalize the state before training, ensuring stable dynamics ranges; (ii) we augment the model input with a fixed window of past k global states and joint actions to provide richer temporal context for next-state prediction.

3.2 Learning Behaviors in Imagination

To support reinforcement learning with imagined rollouts, we pair DIMA with two necessary components. The first is a reward and termination model f_ϕ where reward prediction and termination pre-
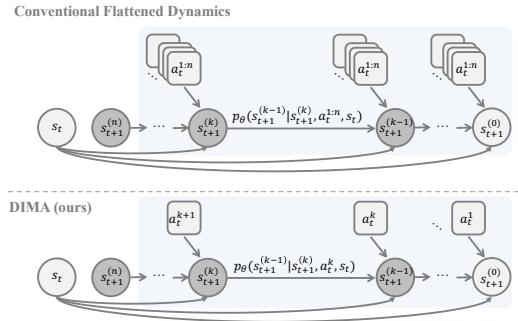
diction are framed as scalar regression and binary classification tasks, respectively. Motivated by the advanced sequence modeling capability of Transformer [28], we employ a Transformer architecture as the backbone. As illustrated in Figure 3, the model takes sequences of $(\ldots, s_t, a_t^{1:n}, s_{t+1}, a_{t+1}^{1:n}, \ldots)$ as input and predicts reward and termination at each timestep via two separate 3-layer multilayer perceptron (MLP) heads on top of the shared output embedding. The model is built upon MinGPT implementation [29]. The second component is a special auto-encoder $g_\varphi(o_t^{1:n}|s_t)$ that encodes the global state s_t into a compact latent space and decodes it into the joint observation $o_t^{1:n}$. We implement this using a simple yet effective VQ-VAE [30] with Finite Scalar Quantization [31]. We adopt an actor-critic framework to learn the behavior policy of each agent, where the actor and critic are parameterized by two 3-layer MLPs, $\pi_\psi(a_t^i|o_t^i)$ and $V_\xi(s_t)$, respectively.
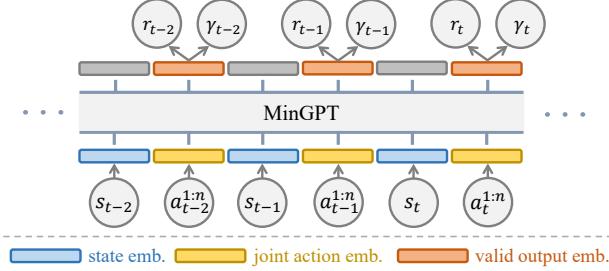

Figure 3: Overview of the reward and termination model. DIMA addresses reward and termination prediction from a global perspective using a transformer architecture to capture temporal correlations. Both functions share the same backbone with separate prediction heads.

Thanks to DIMA's global state transition predictions, we can leverage oracle information from the global state to train a centralized critic, which in turn guides the optimization of decentralized agent actors. This naturally aligns with the centralized training with decentralized execution (CTDE) paradigm commonly used in model-free MARL [32, 33, 34]. Moreover, this provides a clear advantage over recent model-based MARL methods that also rely on learning in imagination [12, 14]. As these methods typically model local observation dynamics for scalability, they lose the benefits of accessing oracle global state, which our approach fully exploits. Here we train the actor and critic with MAPPO [34]. λ-return [1] is used as the target to update the value function. The details of behavior learning objectives and algorithmic description are presented in §C and §F, respectively.

As we evaluate DIMA under the learning in imagination paradigm, our approach iteratively executes a cycle that comprises three steps: (i) collecting experience by executing the policy, (ii) updating the world model with the collected experience, and (iii) learning the policy through imagined rollouts within the learned world model. Note that throughout the whole procedure, the historical experiences stored in the replay buffer are only used for training the world model, while the policy is optimized through unlimited imagined trajectories generated by the world model.

4 Related Works

Diffusion Model for RL. Diffusion models [16, 35] have been applied in reinforcement learning (RL) for its strong generation capability. Specifically, they are capable of modeling complex action distributions in online RL [36, 37, 38, 39], offline policy learning [40, 41, 42], and imitation learning [43, 44, 45]. Other works also adopt diffusion models as planners to generate state-action sequences [46, 47, 48]. Recently, diffusion policies are also used as an action expert to combine with LLMs and obtain visual-language-action model [49, 50]. For dynamics modeling, diffusion models have been employed as alternatives to autoregressive models to learn the complex transition function of MDPs [18, 51, 19], while they are limited in the single-agent domain. MADiff [26] learns the distribution of the whole trajectory in the offline multi-agent settings, without modeling the step-wise transitions. Other works [52, 53, 54] treat diffusion-based world modeling as a video generation problem without taking actions as a condition, limiting their abilities. In contrast, our proposed DIMA predicts future states based on sequential action conditions, which effectively builds the multi-agent world model.

Multi-Agent RL. In cooperative MARL, agents coordinate to maximize a joint reward function. Centralized Training with Decentralized Execution (CTDE) [55] is a foundational framework that leverages the global state of agents during training to facilitate policy learning while relying on partial information during execution. CTDE framework serves the basis for both value-based [32, 33, 56] and policy-based MARL methods [57, 34, 58]. Additionally, some works reformulate MARL as a sequential decision-making problem [59, 60], offering insight into sequential denoising in diffusion-based dynamics. Model-based MARL has gained significant attention for its ability to explicitly model the underlying MDPs in multi-agent environments. Notable examples include MAZero [13], which

adapts MuZero-style planning with MCTS, and Dreamer-based methods [12, 61, 14], which leverage learning in imaginations for multi-agent setups [1, 2, 62]. These approaches have demonstrated the potential of model-based methods to improve coordination in multi-agent systems.

More recently, diffusion models have been introduced into MARL to enhance coordination and trajectory modeling, motivated by their advanced modeling capabilities. MADiff [26] first introduces diffusion models in MARL through offline trajectory learning via attention-based diffusion. Subsequent works have extended the use of diffusion models in MARL. Specifically, DoF [63] investigates offline MARL by factorizing a centralized diffusion model into multiple sub-models, aligning with the CTDE framework. Similarly, MADiTS [27] explores diffusion-based data augmentation by stitching high-quality coordination segments together. While effective, these methods primarily use diffusion models as goal-conditioned trajectory generators, failing to account for the underlying multi-agent dynamics. Our proposed DIMA addresses this research gap by constructing an effective world model that explicitly captures the multi-agent dynamics. By leveraging the strengths of diffusion-inspired modeling, DIMA assists policy training and improves the overall performance of MARL.

5 Experiments

5.1 Experiments Setup

Environments. We evaluate our method on two widely-used multi-agent continuous control benchmarks requiring heterogeneous-agent cooperation: Multi-Agent MuJoCo (MAMuJoCo) [22] and Bimanual Dexterous Hands (Bi-DexHands) [23]. MAMuJoCo extends MuJoCo [64] to multi-agent settings by partitioning a robot into agents controlling different degrees of freedom (DoFs), requiring coordination for coherent movement. We use seven agent-partitioning settings: HalfCheetah [2x3, 3x2, and 6x1]; Walker [2x3 and 3x2]; and Ant [2x4 and 4x2]. Bi-DexHands features dual ShadowRobot hands (26 DoFs each) performing precise bimanual manipulation. We evaluate on four tasks: *ShadowHandPen*, *ShadowHandDoorOpenOutward*, *ShadowHandDoorOpenInward*, and *ShadowHandBottleCap*. To highlight the sample efficiency of learning in imaginations, we adopt a low-data regime [65], limiting real-environment samples to 1M for MAMuJoCo and 300k for Bi-DexHands, adjusted for their different episode lengths. In model-based MARL where policies are learned in imaginations, performance directly reflects the accuracy of the world model, enabling transparent evaluation.

Baselines. We compare DIMA against two strong model-based baselines with the same policy learning paradigm as ours – MAMBA [12] and MARIE [14]. MAMBA extends DreamerV2 [39] to the multi-agent context and establishes an effective Recurrent State Space Model (RSSM)-based world model. MARIE incorporates Transformer-based autoregressive world modeling [3] with CTDE principle and demonstrates remarkable sample efficiency on the benchmark with discrete action space. We also compare DIMA with strong model-free baselines, including two on-policy algorithms MAPPO [34] and HAPPO [58], and an off-policy algorithm HASAC [66, 67]. HASAC is a heterogeneous-agent extension of SAC [68] which is well known for its high sample efficiency. Each algorithm is evaluated using 4 random seeds per scenario. For each random seed, we report the averaged episode return across 10 evaluation episodes at fixed intervals of environment steps. To ensure a fair comparison, we restrict the imagination horizon $H = 15$ for all model-based algorithms. Results of MARIE would not be reported in Bi-DexHands due to severe out-of-memory issues under our available computational resources.

5.2 Main Results

DIMA consistently outperforms all evaluated baselines across a wide range of multi-agent continuous control tasks, achieving superior sample efficiency and higher final returns. As shown in Figure 4 and 5, DIMA exhibits rapid and consistent policy convergence across all chosen MAMuJoCo and Bi-DexHands tasks, while other model-based baselines fail to demonstrate such stable learning behavior. This highlights the advantage of our approach in leveraging an effective world modeling formulation that is better aligned with the global state transitions of the environment. MARIE and MAMBA suffer from a mismatch with the true global transition dynamics inherent in Dec-POMDPs due to their integration of local dynamics modeling with the CTDE principle. This discrepancy potentially imposes an inherent limitation on model accuracy, particularly in environments like MAMuJoCo where inter-agent dependencies are strongly correlated. Although enjoying world modeling complexity at a linear rate, such architectural misalignment limits their
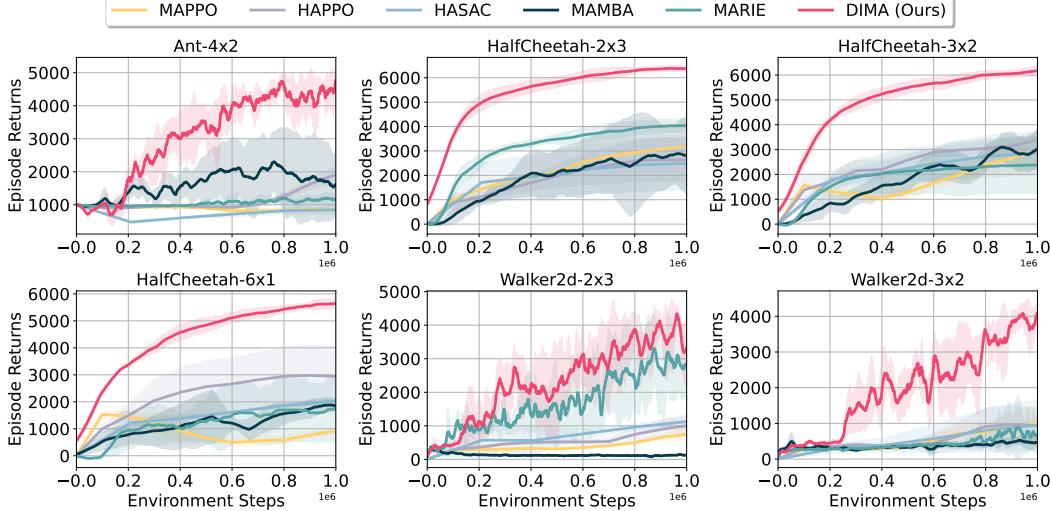
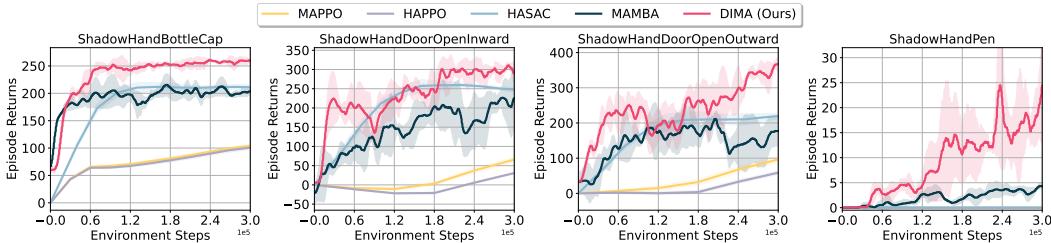Figure 4: Curves of averaged episode returns for all methods in MAMuJoCo.

Figure 5: Curves of averaged episode returns for all methods in Bi-DexHands.

scalability to highly coupled settings. Interestingly, MARIE and MAMBA perform comparably to—or even worse than sample-inefficient on-policy model-free methods HAPPO and MAPPO (e.g., in HalfCheetah [3x2, 6x1]), whereas DIMA consistently demonstrates superior performance. This performance gain reflects the accuracy and robustness of DIMA in enabling more precise and reliable imaginations for policy optimization.

A similar trend is observed in the Bi-DexHands benchmark, characterized by the control of two dexterous hands, each with 26 DoFs (i.e., $a_t^i \in \mathbb{R}^{26}$). Benefiting from the expressiveness of the diffusion model, DIMA is able to more accurately capture especially sophisticated and contact-rich dynamics. By learning a denoising generative process under our formulation, DIMA enables more faithful representations of the underlying transition distribution and leads to more stable, coherent imagined trajectories for downstream policy learning, compared to RSSM-based models. As a result, DIMA substantially improves the learning of dexterous manipulation policies in scenarios requiring fine-grained, high-precision coordination. The numerical results are further provided in §E.

5.3 Model Analysis

DIMA demonstrates substantially more accurate and stable long-horizon predictions than existing multi-agent world models. To better evaluate the model capabilities among MAMBA, MARIE and our DIMA, we visualize their imagined trajectories alongside the ground truth (GT) on Ant [2x4] task. As visualized in Figure 6, DIMA generates a consistent imagined trajectory that closely aligns with the ground truth (GT) across the full prediction horizon $H = 15$, maintaining coherent agent structures and motion patterns. In contrast, MARIE and MAMBA both exhibit significant degradation as the horizon extends, and suffer from varying degrees of distortions. These issues are especially pronounced at challenging future timesteps such as $t = 4$ and $t = 12$, highlighted by red bounding boxes. The qualitative results underscore DIMA's superior modeling capability and stability in capturing complex continuous multi-agent control dynamics, which is critical for generating reliable imagined rollouts that support sample-efficient policy learning.

DIMA effectively preserves permutation invariance over long-horizon multi-agent predictions. To validate whether and how DIMA exhibits the desired *permutation invariance* property elaborated
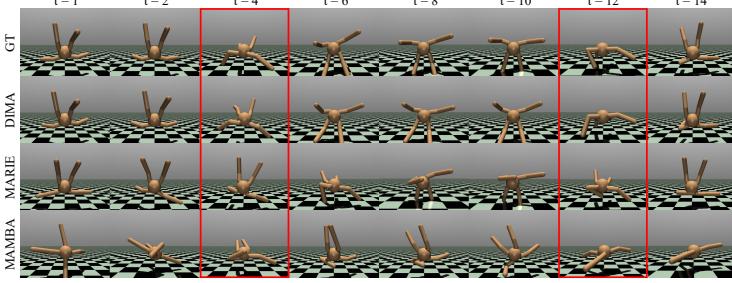
Figure 6: Reconstructions of long-term predictions of different multi-agent world models. We qualitatively compare the reconstruction quality of different multi-agent world models. Each model perform forward imagination over a horizon of $H = 15$.
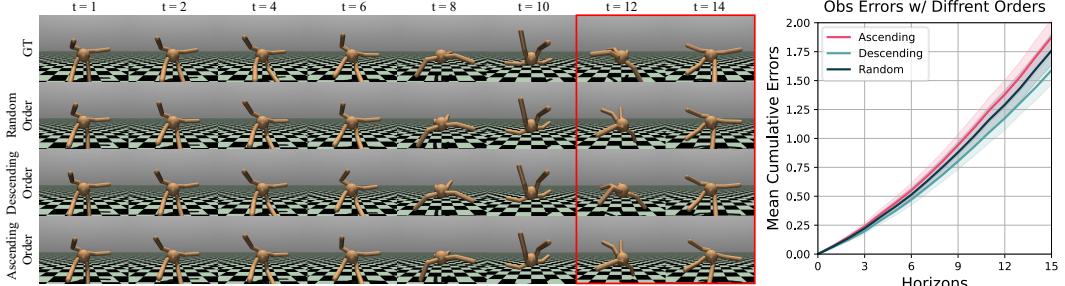

Figure 7: Visualization of long-term predictions with different conditioning orders, together with the accompanying cumulative observation errors curve.

in §3.1, we evaluate its unrolling behavior under different conditioning orders of agent actions. As shown in Figure 7 (left), we generate imagined rollouts from the same initial state and joint action set, but vary the conditioning order using three representative orders: random, ascending and descending w.r.t. agent ids. DIMA produces visually consistent rollouts across different orders until notable visual differences emerges at $t = 12$, highlighted by the red bounding box. This demonstrates that DIMA maintains this consistency effectively up to a long prediction horizon at least $H = 10$. To further quantify the consistency, we plot the mean cumulative observation errors over prediction horizons under each order, as depicted in Figure 7 (right). The resulting curves seems quite aligned, with no significant deviation among the three conditions, indicating that DIMA exhibits the *permutation invariance* property within a considerably long horizon via optimizing Eq. (9). Details of this experiment setup is provided in §E.

5.4 Ablation Study

Our proposed formulation leads to more stable final performance with reduced variance. To evaluate the impact of DIMA's agent-wise sequential modeling formulation, we compare it against the conventional centralized (joint) modeling approach, where information from all agents is injected into the global world model simultaneously. As shown in Figure 8, despite DIMA's reduced modeling complexity, it achieves comparable overall performance to centralized world models while consistently deliver-

Figure 8: Comparison of our proposed formulation against the conventional centralized formulation.

ing lower variance in final performance. This demonstrates DIMA's superior stability in dynamics prediction under the same data budget, resulting in more reliable policies when trained in imagination.

6 Conclusion

This paper presented a multi-agent world model motivated by the conceptual similarity between the progressive denoising process and the incremental reduction of uncertainty in predicting the global next state in MARL. Then, we propose DIMA that models multi-agent dynamics from a centralized perspective while achieving reduced complexity, seamlessly aligning the world model with the underlying MDPs to obtain more accurate predictions. To validate the efficacy of DIMA, we integrated it into the learning-in-imagination training scheme and conducted extensive experiments on the MAMuJoCo and Bi-DexHands benchmarks. The results demonstrated DIMA's superior accuracy
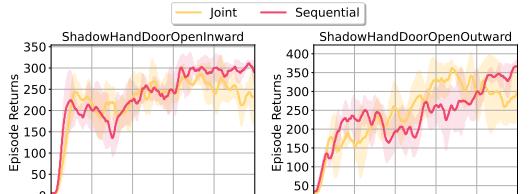
and robustness in predicting environment dynamics, as well as its ability to enhance sample efficiency and overall performance. Despite its effectiveness, DIMA may encounter scalability challenges when applied to large-scale multi-agent systems with hundreds of agents. To address this, we plan to explore grouping techniques to further extend DIMA's applicability and scalability in future work.

References

[1] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1lOTC4tDS`.

[2] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=0oabwyZbOu`.

[3] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=vhFu1Acb0xb`.

[4] Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=TdBaDGCpjly`.

[5] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[6] Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. *Advances in neural information processing systems*, 34:25476–25488, 2021.

[7] Nicklas A Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, pages 8387–8406. PMLR, 2022.

[8] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=Oxh5CstDJU`.

[9] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E. Taylor. A very condensed survey and critique of multiagent deep reinforcement learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020.

[10] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE Transactions on Cybernetics*, 50:3826–3839, 2020. doi: 10.1109/TCYB.2020.2977374.

[11] Frans A Oliehoek, Christopher Amato, et al. A concise introduction to decentralized POMDPs, volume 1. *Springer*, 2016.

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

[13] Qihan Liu, Jianing Ye, Xiaoteng Ma, Jun Yang, Bin Liang, and Chongjie Zhang. Efficient multi-agent reinforcement learning by planning. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=CpnKq3UJwp`.

[14] 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. *arXiv preprint arXiv:2406.15836*, 2024.

[15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[18] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

[19] Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning. *arXiv preprint arXiv:2402.03570*, 2024.

[20] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=P8pqeEkn1H`.

[21] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=k7FuTOWMOc7`.

[22] Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems*, 34:12208–12221, 2021.

[23] Yuanpei Chen, Yaodong Yang, Tianhao Wu, Shengjie Wang, Xidong Feng, Jiechuan Jiang, Zongqing Lu, Stephen Marcus McAleer, Hao Dong, and Song-Chun Zhu. Towards human-level bimanual dexterous manipulation with reinforcement learning. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL `https://openreview.net/forum?id=D29JbExncTP`.

[24] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[25] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[26] Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. Madiff: Offline multi-agent learning with diffusion models. *Advances in Neural Information Processing Systems*, 37:4177–4206, 2024.

[27] Lei Yuan, Yuqi Bian, Lihe Li, Ziqian Zhang, Cong Guan, and Yang Yu. Efficient multi-agent offline coordination via diffusion-based trajectory stitching. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=EpnZEzYDUT`.

[28] 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.

[29] Andrej Karpathy. mingpt: A minimal pytorch re-implementation of the openai gpt (generative pretrained transformer) training. `https://github.com/karpathy/minGPT`, 2020.

[30] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[31] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=8ishA3LxN8`.

[32] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017. URL `https://arxiv.org/abs/1706.05296`.

[33] 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, 2018. URL `https://arxiv.org/abs/1803.11485`.

[34] 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, 2022. URL `https://arxiv.org/abs/2103.01955`.

[35] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications, 2024. URL `https://arxiv.org/abs/2209.00796`.

[36] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=AHvFDPi-FA`.

[37] Linjiajie Fang, Ruoxue Liu, Jing Zhang, Wenjia Wang, and Bingyi Jing. Diffusion actor-critic: Formulating constrained policy iteration as diffusion noise regression for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=ldVkAO09Km`.

[38] Wenhao Li. Efficient planning with latent diffusion. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=btpgDo4u4j`.

[39] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[40] Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

[41] Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Soft diffusion actor-critic: Efficient online reinforcement learning for diffusion policy, 2025. URL `https://arxiv.org/abs/2502.00361`.

[42] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization, 2024. URL `https://arxiv.org/abs/2405.16173`.

[43] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[44] Xiang Li, Varun Belagali, Jinghuan Shang, and Michael S. Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning, 2024. URL `https://arxiv.org/abs/2307.01849`.

[45] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Josh Tenenbaum. Learning to act from actionless videos through dense correspondences. *ArXiv*, abs/2310.08576, 2023. URL `https://api.semanticscholar.org/CorpusID:263908842`.

[46] Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[47] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[48] Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. In *Neural Information Processing Systems*, 2023.

[49] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π0: A vision-language-action flow model for general robot control, 2024. *URL https://arxiv. org/abs/2410.24164*, 2024.

[50] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[51] Marc Rigter, Jun Yamada, and Ingmar Posner. World models via policy-guided trajectory diffusion, 2024. URL `https://arxiv.org/abs/2312.08533`.

[52] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=bo8q5MRcwy`.

[53] Sherry Yang, Yilun Du, Seyed Kamyar Seyed Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=sFyTZEqmUY`.

[54] Siyuan Zhou, Yilun Du, Jiaben Chen, YANDONG LI, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=kHjOmAUfVe`.

[55] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning, 2019. URL `https://arxiv.org/abs/1905.05408`.

[56] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning, 2021. URL `https://arxiv.org/abs/2008.01062`.

[57] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients, 2024. URL `https://arxiv.org/abs/1705.08926`.

[58] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=EcGGFkNTxdJ`.

[59] Dimitri Bertsekas. Multiagent rollout algorithms and reinforcement learning, 2020. URL `https://arxiv.org/abs/1910.00120`.

[60] Jianing Ye, Chenghao Li, Jianhao Wang, and Chongjie Zhang. Towards global optimality in cooperative marl with the transformation and distillation framework, 2023. URL `https://arxiv.org/abs/2207.11143`.

[61] Zifan Wu, Chao Yu, Chen Chen, Jianye Hao, and Hankz Hankui Zhuo. Models as agents: Optimizing multi-step predictions of interactive local models in model-based multi-agent reinforcement learning, 2023. URL `https://arxiv.org/abs/2303.17984`.

[62] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL `https://arxiv.org/abs/2301.04104`.

[63] Chao Li, Ziwei Deng, Chenxing Lin, Wenqi Chen, Yongquan Fu, Weiquan Liu, Chenglu Wen, Cheng Wang, and Siqi Shen. Dof: A diffusion factorization framework for offline multi-agent reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=OTFKVkxSlL`.

[64] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[65] Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model based reinforcement learning for atari. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1xCPJHtDB`.

[66] Yifan Zhong, Jakub Grudzien Kuba, Xidong Feng, Siyi Hu, Jiaming Ji, and Yaodong Yang. Heterogeneous-agent reinforcement learning. *Journal of Machine Learning Research*, 25(32): 1–67, 2024. URL `http://jmlr.org/papers/v25/23-0488.html`.

[67] Jiarong Liu, Yifan Zhong, Siyi Hu, Haobo Fu, QIANG FU, Xiaojun Chang, and Yaodong Yang. Maximum entropy heterogeneous-agent reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=tmqOhBC4a5`.

[68] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[69] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

A Proof of Theorem 2

Proof. Given the log-likelihood of the global state transition, we have the following:

$$
\begin{aligned}
\log P(s_{t+1}|s_t, a_t^{1:N}) &= \log \int p(s_{t+1}, s_{t+1}^{(1):(n)}|s_t, a_t^{1:n})\, ds_{t+1}^{(1):(n)} \\
&= \log \int \frac{p(s_{t+1}, s_{t+1}^{(1):(n)}|s_t, a_t^{(1):(n)})\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}{\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}\, ds_{t+1}^{(1):(n)} \\
&= \log \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}\left[\frac{p(s_{t+1}, s_{t+1}^{(1):(n)}|s_t, a_t^{1:n})}{\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}\right] \\
&\geq \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}\left[\log \frac{p(s_{t+1}, s_{t+1}^{(1):(n)}|s_t, a_t^{1:n})}{\hat{q}(s_{t+1}^{(1):(n)}|s_t, a_t^{1:n}, s_{t+1})}\right],
\end{aligned}
\tag{11}
$$

where the last inequality results from Jensen's inequality. Under the definition and property of the conditional Markovian forward diffusion process $\hat{q}$ in Eqs. (2)–(5) and Assumption 1, we can rewrite Eq. (11) as follows,

$$
\log P(s_{t+1}|s_t, a_t^{1:N}) \geq \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})\prod_{k=1}^{n} p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k))}{\prod_{k=1}^{n}\hat{q}(s_{t+1}^{(k)}|s_{t+1}^{(k-1)})}\right],
\tag{12}
$$

where we denote $s_{t+1} := s_{t+1}^{(0)}$. Then, RHS of Eq. (12) can be further simplified,

$$
\begin{aligned}
&\mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})\prod_{k=1}^{n} p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k))}{\prod_{k=1}^{n}\hat{q}(s_{t+1}^{(k)}|s_{t+1}^{(k-1)})}\right] \\
&= \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, s_t, a_t^1)}{\hat{q}(s_{t+1}^{(1)}|s_{t+1}^{(0)})} + \log \prod_{k=2}^{n} \frac{p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k)}{\hat{q}(s_{t+1}^{(k)}|s_{t+1}^{(k-1)})}\right] \\
&= \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, s_t, a_t^1)}{\hat{q}(s_{t+1}^{(1)}|s_{t+1}^{(0)})} + \log \prod_{k=2}^{n} \frac{p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k)}{\frac{\hat{q}(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})\hat{q}(s_{t+1}^{(k)}|s_{t+1}^{(0)})}{q(s_{t+1}^{(k-1)}|s_{t+1}^{(0)})}}\right] \\
&= \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, s_t, a_t^1)}{\hat{q}(s_{t+1}^{(1)}|s_{t+1}^{(0)})} + \log \frac{\hat{q}(s_{t+1}^{(1)}|s_{t+1}^{(0)})}{\hat{q}(s_{t+1}^{(n)}|s_{t+1}^{(0)})} + \log \prod_{k=2}^{n} \frac{p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k)}{\hat{q}(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})}\right] \\
&= \mathbb{E}_{\hat{q}(s_{t+1}^{(1):(n)}|s_{t+1})}\left[\log \frac{p(s_{t+1}^{(n)})p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, s_t, a_t^1)}{\hat{q}(s_{t+1}^{(n)}|s_{t+1}^{(0)})} + \sum_{k=2}^{n}\log \frac{p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_t, a_t^k)}{\hat{q}(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})}\right]
\end{aligned}
$$

Therefore, the evidence of dynamics transition can be bounded as follows:

$$
\begin{aligned}
\log P(s_{t+1}|s_t, a_t^{1:N}) \geq\; & \mathbb{E}_{\hat{q}(s_{t+1}^{(1)}|s_{t+1}^{(0)})}[\log p(s_{t+1}^{(0)}|s_{t+1}^{(1)}, s_t, a_t^1)] - D_{\mathrm{KL}}[\hat{q}(s_{t+1}^{(n)}|s_{t+1}^{(0)})\|p(s_{t+1}^n)] \\
& - \sum_{k=2}^{n}\mathbb{E}_{\hat{q}(s_{t+1}^{(k)}|s_{t+1}^{(0)})}\left[D_{\mathrm{KL}}(\hat{q}(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, s_{t+1}^{(0)})\|p(s_{t+1}^{(k-1)}|s_{t+1}^{(k)}, a_t^k, s_t))\right].
\end{aligned}
\tag{13}
$$

As shown by [25], the conditional forward diffusion process $\hat{q}$ behaves identically to the unconditional one q. Therefore, we can substitute the $\hat{q}$ with the q in Eq. (13), concluding our proof. □

B EDM Preconditioners and Noise Scheduler

To keep input and output signal magnitudes fixed to the same scale and avoid large variance in gradient magnitudes on a per-sample basis, Karras et al. [21] introduced the following preconditioners for

normalization and re-scaling output to stabilize and improve the training dynamics of the network:

$$c_{\text{in}}^{\tau} = \frac{1}{\sqrt{\sigma(\tau)^2 + \sigma_{\text{data}}^2}} \tag{14}$$

$$c_{\text{out}}^{\tau} = \frac{\sigma(\tau)\sigma_{\text{data}}}{\sqrt{\sigma(\tau)^2 + \sigma_{\text{data}}^2}} \tag{15}$$

$$c_{\text{noise}}^{\tau} = \frac{1}{4}\log(\sigma(\tau)) \tag{16}$$

$$c_{\text{skip}}^{\tau} = \frac{\sigma_{\text{data}}^2}{\sigma_{\text{data}}^2 + \sigma(\tau)^2}, \tag{17}$$

where $\sigma_{\text{data}} = 0.5$ in our experiment hyperparameter setup. The noise scheduler for training the diffusion model follows the same design in [21], described as follows:

$$\sigma(\tau) = \tau, \ \log(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2), \tag{18}$$

where $P_{\text{mean}} = -0.4$ and $P_{\text{std}} = 1.2$.

C Behavior Learning Details

Inspired by the success of MARIE [14], we adopt MAPPO [34] to train both the actor and critic inside the imaginations of DIMA. A key distinction from MARIE is that our model explicitly predicts the global state, enabling seamless integration with CTDE techniques as well as actor–critic architectures commonly used in model-free MARL. Therefore, we implement both the actor ψ and critic ξ with two 3-layer MLPs together with ReLU activation and Layer Normalization, respectively. Similar to off-the-shelf CTDE model-free MARL algorithms, we adopt actor parameter sharing across agents.

Critic loss function. We utilize λ-return in DreamerV1 [1], which employs an exponentially weighted average of different k-steps TD targets to balance bias and variance as the regression target for the critic. Given an imagined trajectory $\{\hat{s}_t, \hat{o}_t^{1:n}, a_t^{1:n}, \hat{r}_t, \hat{\gamma}_t\}_{t=1}^H$ over all agents, λ-return is calculated recursively as,

$$V_\lambda(\hat{s}_t) = \hat{r}_t^i + \hat{\gamma}_t \cdot \begin{cases} (1-\lambda)V_\xi(\hat{s}_t) + \lambda V_\lambda(\hat{s}_{t+1}) & \text{if} \quad t < H \\ V_\xi(\hat{s}_t) & \text{if} \quad t = H \end{cases} \tag{19}$$

The objective of the critic ξ is to minimize the mean squared difference $\mathcal{L}_\xi$ with λ-returns over imagined trajectories, as

$$\mathcal{L}_\xi = \mathbb{E}_{\pi_\psi}\left[\sum_{t=1}^{H-1}\left(V_\xi(\hat{s}_t) - \text{sg}\big(V_\lambda(\hat{s}_t)\big)\right)^2\right], \tag{20}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operation. We optimize the critic loss with respect to the critic parameters ξ using the Adam optimizer.

Actor loss function. The objective for the actor $\pi_\psi^i(\cdot|\hat{o}_t^i) := \pi_\psi(\cdot|\hat{o}_t^i)$ is to output actions that maximize the prediction of long-term future rewards made by the critic. To incorporate intermediate rewards more directly, we train the actor to maximize the same λ-return that was computed for training the critic. In terms of the non-stationarity issue in multi-agent scenarios, we adopt PPO updates, which introduce importance sampling for actor learning. The actor loss function for agent i is:

$$\mathcal{L}_\psi^i = -\mathbb{E}_{\pi_{\psi_{\text{old}}}^i}\left[\sum_{t=0}^{H-1}\min\left(r_t^i(\psi)A_t, \text{clip}(r_t^i(\psi), 1-\epsilon, 1+\epsilon)A_t\right) + \eta\mathcal{H}(\pi_\psi^i(\cdot|\hat{o}_t^i))\right] \tag{21}$$

where $r_t^i(\psi) = \pi_\psi^i/\pi_{\psi_{\text{old}}}^i$ is the policy ratio and $A_t = \text{sg}(V_\lambda(\hat{s}_t) - V_\xi(\hat{s}_t))$ is the advantage. Unlike MAPPO, we choose not to design agent-specific global states, as such designs are overly hand-crafted and inject task-specific human priors, which undermines the generality and soundness of the approach. Instead, we retain the environment's original agent-agnostic global state shared among all agents, and feed it into the value function V_ξ. As a result, the estimated advantage function A_t is also shared across all agents during actor updates. We optimize the actor loss with respect to the actor parameters ψ using the Adam optimizer. Overall hyperparameters are shown in Table 1.

Table 1: Behaviour learning hyperparameters.

Hyperparameter	Value
Common	
Imagination horizon (H)	15
λ	0.95
Clipping parameter ϵ	0.1
MAMuJoCo	
Discount factor γ	0.99
η	0.001
Bi-DexHands	
Discount factor γ	0.95
η	0.01

D   Illustrations of Experimental Environments

ShadowHandPen ShadowHandBottleCap ShadowHandDoorOpenInward HalfCheetah 6-agent partitioning
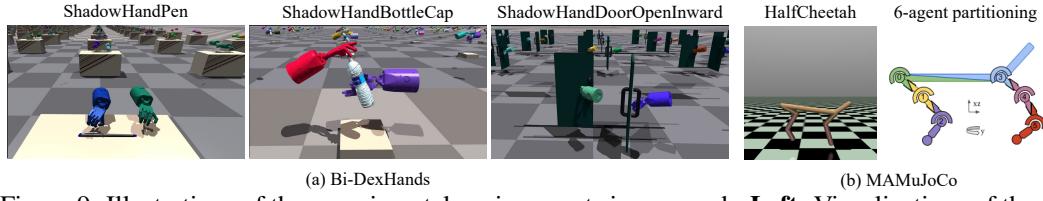

(a) Bi-DexHands (b) MAMuJoCo

Figure 9: Illustrations of the experimental environments in our work. **Left**: Visualizations of three Bi-DexHands tasks: removing a pen cap, opening a bottle cap, and opening a door inwards. **Right**: Visualization of 6-agent partitioning w.r.t. HalfCheetah in MAMuJoCo.

E   Additional Results

E.1   Final Returns of All Methods on MAMuJoCo and Bi-DexHands

Table 2: **Comparison of final episode returns across MAMuJoCo and Bi-DexHands benchmarks.** We report the mean final episode return and standard deviation over 4 random seeds. DIMA consistently outperforms all baselines across all chosen tasks on both MAMuJoCo and Bi-DexHands. The best result per task is highlighted in bold and shaded in blue color, while the second-best is underlined.

Tasks	Steps	Methods					
		DIMA (Ours)	MARIE	MAMBA	HASAC	HAPPO	MAPPO
MAMuJoCo							
Ant-2x4		$4881_{\pm756}$	$\underline{4471}_{\pm553}$	$1314_{\pm756}$	$1344_{\pm282}$	$1716_{\pm449}$	$859_{\pm47}$
Ant-4x2		$4766_{\pm450}$	$1173_{\pm136}$	$1618_{\pm931}$	$850_{\pm126}$	$\underline{1917}_{\pm253}$	$854_{\pm41}$
HalfCheetah-2x3	1M	$6370_{\pm121}$	$\underline{4045}_{\pm275}$	$2813_{\pm1580}$	$2499_{\pm1081}$	$2628_{\pm893}$	$3196_{\pm75}$
HalfCheetah-3x2		$6175_{\pm212}$	$2380_{\pm1145}$	$3029_{\pm798}$	$2872_{\pm890}$	$\underline{3402}_{\pm317}$	$2936_{\pm766}$
HalfCheetah-6x1		$5643_{\pm163}$	$1738_{\pm1213}$	$1848_{\pm220}$	$2044_{\pm110}$	$\underline{2939}_{\pm1113}$	$925_{\pm121}$
Walker2d-2x3		$3329_{\pm1056}$	$\underline{2822}_{\pm997}$	$124_{\pm19}$	$1135_{\pm210}$	$1007_{\pm282}$	$752_{\pm216}$
Walker2d-3x2		$4084_{\pm357}$	$604_{\pm349}$	$466_{\pm103}$	$958_{\pm715}$	$932_{\pm513}$	$\underline{1004}_{\pm480}$
Bi-DexHands							
BottleCap		$259.9_{\pm4.1}$	-	$203.8_{\pm5.2}$	$\underline{210.9}_{\pm6.1}$	$100.7_{\pm3.8}$	$104.0_{\pm2.3}$
DoorOpenInward	300K	$290.4_{\pm29.0}$	-	$225.0_{\pm79.4}$	$\underline{246.3}_{\pm7.0}$	$30.7_{\pm2.5}$	$65.8_{\pm6.9}$
DoorOpenOutward		$367.1_{\pm19.4}$	-	$177.4_{\pm43.1}$	$\underline{221.9}_{\pm7.3}$	$58.8_{\pm4.6}$	$96.4_{\pm8.5}$
BottleCap		$24.4_{\pm11.4}$	-	$\underline{4.3}_{\pm0.4}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$

E.2   Experiment Details of Imagination Evaluation across Different Conditioning Orders

To evaluate DIMA's imagination under different conditioning orders on the 2-agent Ant [2×4] task, we collect 10 episodes by using the final policy induced by our algorithm, and randomly sample 100 trajectory segments to form our trajectory segment dataset. For each segment, we generate imagined rollouts using DIMA with different action conditioning orders.

As the EDM framework decouples inference-time sampling from training, the number of denoising steps need not match the number of agents. Thus, we set the number of denoising steps equal to 4, i.e., twice the number of agents. Letting the agent set be $\{1, 2\}$, we consider three conditioning orders: (i) random order: $(2, 1, 1, 2)$, (ii) ascending order w.r.t. agent id: $(1, 1, 2, 2)$, and (iii) descending order w.r.t. agent id: $(2, 2, 1, 1)$.

To provide a quantitative evaluation in Figure 7 (right), we compute the L1 error per observation dimension at each timestep between the 100 sampled trajectory segments and their corresponding imagined rollouts, and accumulate the errors over the prediction horizon. All observation L1 errors are averaged across 2 agents.

F   Overview of DIMA with Learning in Imaginations

We summarize the overall training procedure of DIMA paired with learning in imaginations in Algorithm 1 below. We denote as $\mathcal{D}$ the replay databuffer which stores data collected from the real environment.

G   Training Details and Hyperparameters

G.1   Model Architecture Details

State decoder.   To enable decentralized execution of policies trained within DIMA's imagination rollouts, each agent must make decisions based solely on its local observation rather than the shared

global state. To support such policy learning, we introduce a necessary state decoder that maps the global state s_t into the corresponding joint local observations $o_t^{1:n}$.

Due to our online model-based MARL setup, the state decoder must be continually updated under a non-stationary data distribution which also shifts continually. Using a vanilla MLP as the state decoder in this setting may lead to issues such as overfitting or mode collapse. To mitigate these risks, we incorporate additional regularization into the decoder design by adopting a Vector Quantized Variational Autoencoder (VQ-VAE) [30], which enforces a compact latent codebook representation via vector quantization. Among various VQ-VAE variants, we choose Finite Scalar Quantization (FSQ) [31] as our final implementation as it removes any auxiliary losses and achieves remarkably high codebook utilization, which indicates its strong and effective regularization.

Our implementation is based on the open-source repository: `https://github.com/lucidrains/vector-quantize-pytorch`. We simply build the encoder E_φ and decoder D_φ as MLPs to deal with continuous non-vision global states and joint local observations. The decoder is designed with the same hyperparameters as the encoder. The loss function for learning the autoencoder is as follows:

$$\mathcal{L}_{\text{FSQ}}(E_\varphi, D_\varphi) = \mathbb{E}_{(s_t, o_t^{1:n}) \sim \mathcal{D}} \left[\| o_t^{1:n} - D_\varphi(E_\varphi(s_t) + \text{sg}(\text{round}(f(E_\varphi(s_t)))) - E_\varphi(s_t)) \|^2 \right], \tag{22}$$

where f is a bounding function such that i-th channel/entry in $\hat{z}_t = \text{round}(f(E_\varphi(s_t)))$ takes one of L_i unique values (here $f : z \to \lfloor L_i/2 \rfloor \tanh(z)$ for i-th channel/entry) and round is the operation to map real-valued inputs to the nearest integers. Therefore, we have an implicit codebook $\mathcal{C}$ with $|\mathcal{C}| = \prod_{i=1}^d L_i$. After training the VAE, our state decoder can be expressed by $g_\varphi(o_t^{1:n} | s_t) = D_\varphi(\text{round}(f(E_\varphi(s_t))))$. The hyperparameters are listed in Table 3.

Diffusion model for dynamics modeling. We use the 1-D variant adapted from the U-Net 2D in DIAMOND [18] as the backbone of the diffusion model D_θ. To predict the next state s_{t+1}, the diffusion model D_θ is initially conditioned on the current global s_t, joint action $a_t^{1:n}$ and the diffusion time τ. To improve next-global-state prediction accuracy, we empirically augment the temporal context by additionally incorporating the last 2 global states and joint actions, extending it from s_t and $a_t^{1:n}$ to $s_{t-2:t}$ and $a_{t-2:t}^{1:n}$. Note that the effect of sequential denoising is confined to the joint action $a_t^{1:n}$ conditioning at the current timestep t, and does not extend to the past joint actions.

Inspired by the success of DIAMOND, we directly adopt the same conditioning mechanism in DIAMOND, and use temporal stacking for global state conditioning and adaptive group normalization for joint action and diffusion time conditioning. The hyperparameters are listed in Table 3.

Transformer as reward and termination model. The Transformer for predicting the reward and termination is built upon the implementation of minGPT [29]. Given a fixed imagination horizon H, it first takes a sequence of length $2H$ composed of global states and joint actions $(\ldots, s_t, a_t^{1:n}, \ldots)$, and encodes every single global state and joint action into d_e-dimensional embedding tensor via 2 separate encoding functions. Then the sequence tensor of shape $2H \times d_e$ is forwarded through fixed Transformer blocks. Finally, the Transformer predicts reward and termination via two separate 3-layer MLP heads $f_\phi(r_t | s_{\leq t}, a_{\leq t}^{1:n})$ and $f_\phi(\gamma_t | s_{\leq t}, a_{\leq t}^{1:n})$, respectively. In general, the loss function is described by

$$\mathcal{L}_\phi = \mathbb{E} \left[\sum_{t=1}^H -\log f_\phi(r_t | s_{\leq t}, a_{\leq t}^{1:n}) - \log f_\phi(\gamma_t | s_{\leq t}, a_{\leq t}^{1:n}) \right]. \tag{23}$$

But in practice, we optimize the reward prediction with a smooth L1 loss function and the termination prediction with a cross-entropy loss function. The hyperparameters are listed in Table 3.

G.2 Computational Resources Used for Training

All our experiments including the evaluation of chosen baselines are run on a machine with a single NVIDIA RTX 4090 GPU, a 24-core CPU, and 256GB RAM.

G.3 Baseline Implementation Details

In our experiments, we reran and evaluated all baseline methods. To ensure fairness for comparisons, we followed the optimal hyperparameters provided in their official implementations, listed below:

Table 3: Architecture details.

Hyperparameter	Value
State Decoder (g_φ)	
MLP layers	3
Hidden size	512
Activation	GELU [69]
FSQ Levels L_i	[8, 6, 5]
Diffusion Model (D_θ)	
Global state conditioning mechanism	Temporal stacking
Joint action conditioning mechanism	Adaptive Group Normalization
Diffusion time conditioning mechanism	Adaptive Group Normalization
Residual blocks layers	[2, 2, 2]
Residual blocks channels	[64, 64, 64]
Residual blocks conditioning dimension	256
Reward and Termination Model (f_ϕ)	
Embedding dimension d_e	256
Transformer block layers	6
Attention heads	4
Embedding dropout	0.1
Attention dropout	0.1
Residual dropout	0.1

- MARIE: `https://github.com/breez3young/MARIE`;
- MAMBA: `https://github.com/jbr-ai-labs/mamba`;
- HASAC, HAPPO and MAPPO: `https://github.com/PKU-MARL/HARL`.

G.4   DIMA hyperparameters

We list the hyperparameters of DIMA paired with learning in imaginations in Table 4.

Table 4: Hyperparameters for DIMA.

Hyperparameter	Value
Batch size for State Decoder training	256
Batch size for Diffusion Model training	64
Batch size for Reward and Termination Model training	128
Optimizer for State Decoder	AdamW
Optimizer for Diffusion Model	AdamW
Optimizer for Reward and Termination Model	AdamW
Optimizer for Actor & critic	Adam
Learning rate for State Decoder	0.0003
Learning rate for Diffusion Model	0.0001
Learning rate for Reward and Termination Model	0.0001
Learning rate for Actor & critic	0.0005
Gradient clipping for State Decoder	10
Gradient clipping for Diffusion Model	1
Gradient clipping for Reward and Termination Model	10
Gradient clipping for Actor & critic	10
Weight decay for State Decoder	0.01
Weight decay for Diffusion Model	0.01
Weight decay for Reward and Termination Model	0.01
λ for λ-return computation	0.95
Discount factor γ	see Table 1
Entropy coefficient	see Table 1
Buffer size (transitions)	2.5×10^5
Training steps per epoch	200
Training steps per epoch for policy learning	4
Sampling Environment steps per epoch	200 in MAMuJoCo 500 in Bi-DexHands
PPO epochs	5
PPO Clipping parameter ϵ	0.1
Number of imagined rollouts	600
Imagination horizon H	15
Diffusion sampling solver	Euler
Number of denoising steps	$\begin{cases} 2 \cdot \|\mathcal{N}\| \text{ if } \|\mathcal{N}\| \leq 2 \\ \|\mathcal{N}\| \text{ if } \|\mathcal{N}\| > 2 \end{cases}$

Algorithm 1: DIMA paired with learning in imaginations

Procedure `training_loop()`:
    for *epochs* **do**
        `collect_experience(`*steps_collect*`)`
        for *steps_state_decoder* **do**
            `update_state_decoder()`
        for *steps_diffusion_model* **do**
            `update_diffusion_model()`
        for *steps_reward_end_model* **do**
            `update_reward_end_model()`
        for *steps_actor_critic* **do**
            `update_actor_critic()`

Procedure `collect_experience(`n`)`:
    $s_0, o_0^{1:n} \leftarrow$ `env.reset()`
    for $t = 0$ **to** $n - 1$ **do**
        Sample $a_t^i \sim \pi_\psi^i(a_t^i|o_t^i)$, $\forall$ agent i
        $s_{t+1}, o_{t+1}^{1:n}, r_t, \gamma_t \leftarrow$ `env.step(`$a_t^{1:n}$`)`
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{s_{t+1}, o_{t+1}^{1:n}, a_t^{1:n}, r_t, \gamma_t\}$
        if $\gamma_t = 1$ **then**
            $s_{t+1}, o_{t+1}^{1:n} \leftarrow$ `env.reset()`

Procedure `update_state_decoder()`:
    Sample state-observation pair $(s_t, o_t^{1:n}) \sim \mathcal{D}$
    Compute $\mathcal{L}_{\text{FSQ}}$ in Eq. (22)
    Update State Decoder g_φ

Procedure `update_diffusion_model()`:
    Sample sequence $\big(s_{t-L+1}, a_{t-L+1}^{1:n}, \ldots, s_t, a_t^{1:n}, s_{t+1}\big) \sim \mathcal{D}$
    Sample $\log(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$ and get $\tau = \sigma$ since $\sigma(\tau) := \tau$
    Sample $s_{t+1}^\tau \sim \mathcal{N}(x_{t+1}^0, \sigma^2 \mathbf{I})$
    Sample a chosen agent id $k \sim \text{Uniform}\{1, 2, \ldots, n\}$
    Compute $\hat{s}_{t+1}^{(0)} = D_\theta(s_{t+1}^\tau; \tau, a_t^k, \underbrace{s_{t-L+1:t}, a_{t-L+1:t-1}^{1:n}}_{\text{extra temporal context}})$
    Compute loss $\mathcal{L}(\theta) = \|\hat{s}_{t+1}^{(0)} - s_{t+1}\|^2$ in Eq. (9)
    Update Diffusion Model D_θ

Procedure `update_reward_end_model()`:
    Sample sequence $\big(s_t, a_t^{1:n}, r_t, \gamma_t, \ldots, s_{t+H-1}, a_{t+H-1}^{1:n}, r_{t+H-1}, \gamma_{t+H-1}\big) \sim \mathcal{D}$
    for $i = t$ **to** $t + H - 1$ **do**
        Compute $\hat{r}_i \sim f_\phi(\hat{r}_i|s_{\leq i}, a_{\leq i}^{1:n})$ and $\hat{\gamma}_i \sim f_\phi(\hat{\gamma}_i|s_{\leq i}, a_{\leq i}^{1:n})$
    Compute $\mathcal{L}_\phi = \sum_{i=t}^{t+H-1} \text{CrossEntropy}(\hat{\gamma}_i, \gamma_i) + \text{SmoothL1}(\hat{r}_i, r_i)$ corresponding to Eq. (23)
    Update Reward and Termination Model f_ϕ

Procedure `update_actor_critic()`:
    Set the joint action condition order $\rho = (i_1, i_2, \ldots, i_n)$
    Sample starting point $\big(s_{t-L+1}, o_{t-L+1}^{1:n}, a_{t-L+1}^{1:n}, \ldots, s_t, o_t^{1:n}\big) \sim \mathcal{D}$ of the imagination
    Let $\hat{o}_t^{1:n} = o_t^{1:n}$
    for $i = t$ **to** $t + H - 1$ **do**
        Sample $a_i^j \sim \pi_\psi^j(a_i^j|\hat{o}_i^j)$ $\forall$ agent j
        Sample the reward $\hat{r}_i$ and the termination $\hat{\gamma}_i$ with f_ϕ
        Sample the next global state $\hat{s}_{t+1}$ by iteratively denoising with D_θ and ρ
        Sample the next joint observation state $\hat{o}_{t+1}^{1:n}$ with g_φ
    Update actor π_ψ^i and critic V_ξ via $\mathcal{L}_\xi$ and $\mathcal{L}_\psi^i$ over imaginations $\{\hat{s}_i, \hat{o}_i^{1:n}, a_i^{1:n}, \hat{r}_i, \hat{\gamma}_i\}_{i=t}^{t+H-1}$