Empowering Multi-Robot Cooperation via Sequential World Models

Zijie Zhao[1,2], **Honglei Guo**[2], **Shengqian Chen**[2], **Kaixuan Xu**[1,2], **Bo Jiang**[2,1],
Yuanheng Zhu[2,1,*], **Dongbin Zhao**[2,1]
[1]School of Artificial Intelligence, University of Chinese Academy of Sciences
[2]Institute of Automation, Chinese Academy of Sciences

ABSTRACT

Model-based reinforcement learning (MBRL) has shown significant potential in robotics due to its high sample efficiency and planning capability. However, extending MBRL to physical multi-robot cooperation remains challenging due to the complexity of joint dynamics and the reliance on synchronous communication. To address this, we propose the **Seq**uential **W**orld **M**odel (SeqWM), a novel framework that integrates the sequential paradigm into model-based multi-agent RL. SeqWM employs independent, autoregressive agent-wise world models to represent joint dynamics, where each agent generates its future trajectory and plans its actions based on the predictions of its predecessors. This design lowers modeling complexity, alleviates the reliance on communication synchronization, and enables the emergence of advanced cooperative behaviors through explicit intention sharing. Experiments in challenging simulated environments (Bi-DexHands and Multi-Quad) demonstrate that SeqWM outperforms existing state-of-the-art model-based and model-free baselines in both overall performance and sample efficiency, while exhibiting advanced cooperative behaviors such as predictive adaptation, temporal alignment, and role division. Furthermore, SeqWM has been successfully deployed on physical quadruped robots, validating its effectiveness in real-world multi-robot systems. Demos and code are available at: SeqWM-MARL.

1 INTRODUCTION

Model-based reinforcement learning (MBRL) has been widely applied to robotic systems due to its high sample efficiency (Wu et al., 2023) and planning capability (Sun et al., 2023). However, extending MBRL to multi-robot cooperation remains challenging. Early decentralized model-based multi-agent reinforcement learning (MARL) approaches built independent world models for each agent (Egorov & Shpilman, 2022), overlooking inter-agent couplings and hindering coordination. More recent centralized methods, by contrast, assume full observability or unrestricted communication (Chai et al., 2024; Toledo, 2024; Liu et al., 2024b), performing dynamics modeling and policy optimization in the joint space. These methods face challenges related to modeling complexity and synchronous communication in robotic systems with high-dimensional observation and action spaces, limiting their deployment in real-world scenarios.

Between centralized and decentralized paradigms, the distributed sequential paradigm has rapidly developed in recent years and demonstrated unique advantages (Khan, 2025). It reformulates multi-agent decision-making as an autoregressive process: agents communicate and act in a certain order, with each updating its policy conditioned on messages and actions from predecessors (Wen et al., 2022; Hu et al., 2025). This design enables more consistent joint reasoning (Ding et al., 2024) and finer-grained credit assignment (Kuba et al., 2022; Wang et al., 2023) without relying on full communication. From a real-world deployment
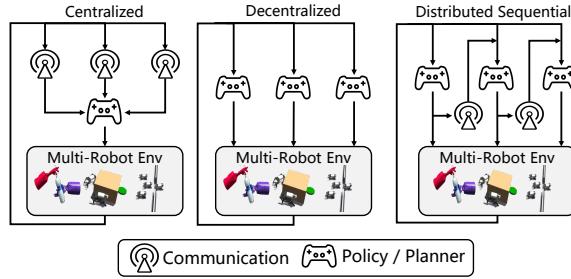

Figure 1: Comparison of SeqWM's distributed sequential paradigm with existing centralized/decentralized paradigms.

perspective, it reduces the reliance on communication synchronization and offers improved robustness against packet loss or disturbances (Ding et al., 2024).

Motivated by these advantages, as shown in Figure 1, we propose the **Seq**uential **W**orld **M**odel (SeqWM), which integrates the sequential paradigm into model-based MARL to structurally decompose dynamics modeling and action planning. For trajectory prediction, SeqWM represents the joint dynamics as sequential agent-wise rollouts following the communication order, where each agent maintains an independent world model and conditions on the predicted trajectories and actions of its predecessors. For action planning, each agent performs multi-step lookahead conditioned on its predecessors' predictions, thereby preserving cooperative performance while constraining the search to a low-dimensional subspace consistent with the sequential structure. We evaluate SeqWM in two challenging multi-robot cooperation environments: Bi-DexHands (Chen et al., 2024a) and Multi-Quad (Xiong et al., 2024), and further validate its effectiveness on physical multi-robot tasks using two Unitree Go2-W robots. The key contributions are as follows:

(1) By integrating the sequential paradigm, SeqWM decomposes joint dynamics into autoregressive agent-wise models, reducing modeling complexity and the need for synchronized communication, thereby extending model-based MARL to multi-robot cooperation.

(2) Through explicit intention sharing, SeqWM enables the emergence of advanced cooperative behaviors such as predictive adaptation, temporal alignment, and role division.

(3) SeqWM achieves state-of-the-art performance in both simulation and real-world quadruped experiments, notably becoming the first multi-agent method to succeed on `BottleCap` and `PushBox`.

These results collectively demonstrate that SeqWM, by leveraging sequential world modeling and planning, offers an effective pathway for multi-robot cooperation, balancing performance, efficiency, and real-world applicability.

2 RELATED WORK

Model-based RL. In single-agent robotics, model-based RL has shown remarkable success due to its high sample efficiency (Zhou et al., 2024; Lancaster et al., 2024), with approaches such as PlanCP (Sun et al., 2023) and GPC (Qi et al., 2025) leveraging learned dynamics models to predict trajectories and optimize actions for physical robots. In contrast, existing model-based MARL methods often rely on centralized paradigms (Zhao et al., 2025b), hindering their practical deployment in multi-robot systems. For example, MACD (Chai et al., 2024) and CoDreamer (Toledo, 2024) use transformers or GNNs to integrate full state-action across all agents. Recent efforts such as MARIE (Zhang et al., 2025a) explored decentralized dynamics modeling, but still require communication at each prediction step for agent-wise aggregation. Different from these works, SeqWM assigns each agent an independent world model and predicts trajectories sequentially, which not only lowers modeling complexity but also substantially reduces the reliance on synchronous communication, thereby making it applicable to real-world scenarios.

Sequential Paradigm. Recent studies in MARL have highlighted the advantages of the sequential paradigm (Khan, 2025), which enables fine-grained credit assignment (Kuba et al., 2022), efficient dynamics modeling (Zhang et al., 2025b), and scalable coordination (Xu et al., 2025). For example, MAT (Wen et al., 2022) and PMAT (Hu et al., 2025) model the multi-agent decision-making process as a sequence prediction problem, employing transformers to autoregressively predict each agent's actions. A2PO (Wang et al., 2023) and HARL (Liu et al., 2024a; Zhong et al., 2024) further introduce the sequential update scheme that bring clearer interpretability and ensure monotonic improvement. SeqComm (Ding et al., 2024) extends this idea to the communication domain, where agents exchange information in a sequential order, effectively mitigating non-stationarity. Motivated by these benefits, we integrate the sequential paradigm into model-based MARL, leveraging world models to enhance planning and coordination.

3 PRELIMINARIES

Problem Formulation. We model the fully cooperative task as a decentralized partially observable Markov decision process (dec-POMDP) (Zhao et al., 2025a), $\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, \mathcal{O}, \mathcal{A}, \Omega, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where

$\mathcal{I} = \{v^1, \ldots, v^n\}$ is the set of agents, $\mathcal{S}$ is the global state space, $\mathcal{O} = \prod_{i=1}^{n} O^i$ is the joint observation space, and $\mathcal{A} = \prod_{i=1}^{n} A^i$ is the joint action space. The observation function $\Omega : \mathcal{S} \times \mathcal{I} \to \mathcal{O}$ defines each agent's perception of the environment, while the transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ specifies the environment dynamics. The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ provides a shared scalar signal, and γ is the discount factor. Each agent v^i learns a local policy $\pi^i : O^i \to A^i$, which maps its observation o^i to an action a^i. The objective is to learn a joint policy $\pi = \prod_{i=1}^{n} \pi^i$ that maximizes the expected discounted return $\sum_{\tau=t}^{\infty} \gamma^\tau r_\tau$.

Sequential Communication and Decision-Making. In many real-world applications, multi-robot systems are often distributed rather than fully decentralized (Negenborn & Maestre, 2014), allowing inter-agent communication to enhance cooperative performance. A Dec-POMDP can thus be extended to a multi-agent POMDP (Oliehoek et al., 2016), where each agent v^i receives messages e_t^i from other agents and updates its policy $\pi^i : O^i \times E \to A^i$. To balance efficiency and decision quality, agents adopt communication protocols (Yu et al., 2023), defined as $\phi^i : O^i \times E \times A^i \to E$. Among them, sequential communication is especially popular for its simplicity and effectiveness (Ding et al., 2024). It organizes agents in a certain order, where each agent acts on its own observation and the message from its predecessor, then passes information forward. Formally, the process is defined as:

$$a_t^i = \pi^i(o_t^i, e_t^i), \quad e_t^{i+1} = \phi^i(e_t^i, o_t^i, a_t^i), \tag{1}$$

Such a sequential structure naturally motivates us to design a world model that predicts trajectories in the same manner, enabling efficient multi-agent planning.

4 METHODOLOGY

In this section, we propose **SeqWM**, a **Seq**uential **W**orld **M**odel for multi-robot cooperation. Unlike existing centralized or decentralized multi-agent world models, SeqWM decomposes the joint dynamics into agent-wise models arranged in a sequence. This design substantially reduces modeling complexity and the reliance on synchronous communication, enabling deployment in physical multi-robot systems.

4.1 SEQUENTIAL MULTI-AGENT WORLD MODEL

Decomposed Dynamics Modeling. At each timestep t, the observation-action pair of a single agent (o_t^i, a_t^i) can be regarded as a token, and the entire multi-agent system as a sequence of such tokens. This perspective reformulates multi-agent joint dynamics as a sequence modeling problem: given the token sequence $[(o_t^1, a_t^1), \ldots, (o_t^n, a_t^n)]$, the model generates the next-step outcomes $[(o_{t+1}^1, r_{t+1}^1), \ldots, (o_{t+1}^n, r_{t+1}^n)]$. Unlike existing centralized world models (Chai et al., 2024; Zhang et al., 2025a), which assume full communication and fuse all tokens simultaneously for prediction, as shown in Figure 1, our method adopts an autoregressive paradigm. In this setup, agent v^1 first predicts (o_{t+1}^1, r_{t+1}^1) from its local information (o_t^1, a_t^1), and passes the result to v^2. Subsequently, each agent v^i conditions on its own observation–action pair (o_t^i, a_t^i) together with the predictions of all predecessors $\{(o_{t+1}^j, r_{t+1}^j)\}_{j<i}$ to produce (o_{t+1}^i, r_{t+1}^i). Such a sequential design reduces communication frequency and modeling complexity in a structured, scalable manner, making the approach well-suited for real-world deployment.

Multi-Agent World Model. As noted in INTRODUCTION, multi-robot cooperative tasks involve high-dimensional observation and action spaces (Zhu et al., 2025a), making it unsuitable to use reconstructing raw observations as the learning objective of the world model (Hafner et al., 2025). Therefore we remove the explicit decoder and instead perform dynamics prediction entirely in a latent space. To facilitate distributed deployment, each agent maintains an independent world model without parameter sharing. Let z_t^i denote the latent state of agent v^i at timestep t, the world model can be defined as follows:

$$
\begin{aligned}
\text{Encoder:} \quad & z_t^i &&= E^i\left(o_t^i\right) \\
\text{Dynamics:} \quad & \hat{z}_{t+1}^i &&= D^i\left(z_t^i, a_t^i, e_t^i\right) \\
\text{Reward:} \quad & \hat{r}_{t+1}^i &&= R^i\left(z_t^i, a_t^i, e_t^i\right) \\
\text{Communication:} \quad & e_t^{i+1} &&= e_t^i \oplus \left(z_t^i, a_t^i\right) \\
\text{Critic:} \quad & \hat{q}_t^i &&= Q^i\left(z_t^i, a_t^i, e_t^i\right) \\
\text{Actor:} \quad & \hat{a}_t^i &&= \pi^{i,\text{Act}}\left(z_t^i, e_t^i\right).
\end{aligned}
\tag{2}
$$

3

All modules in SeqWM are implemented using MLPs, ensuring architectural simplicity and consistency. For communication function, we adopt concatenation operator $\oplus$ to facilitate modular training.[1] As shown in Section 5.5, this concise design achieves more stable training performance compared to alternatives such as cross-attention and recurrent neural networks (RNNs).

Learning Objective. Let θ_E, θ_D, θ_R, θ_Q, and ψ denote the parameters of the encoder, dynamics predictor, reward predictor, critic, and actor, respectively. Following the self-supervised training framework (Hafner et al., 2025; Hansen et al., 2024), the loss functions can be defined as:

$$\mathcal{L}^i(\theta) = \sum_t^H \lambda^t \left(\underbrace{\left\| \hat{z}_{t+1}^i - \text{sg}(z_{t+1}^i) \right\|^2}_{\text{dynamics loss for } \theta_D, \theta_E} + \underbrace{\text{Soft-CE}\left(\hat{r}_t^i, r_t\right)}_{\text{reward loss for } \theta_R, \theta_E} + \underbrace{\text{Soft-CE}\left(\hat{q}_t^i, G_t\right)}_{\text{Q Loss for } \theta_Q, \theta_E} \right), \tag{3}$$

where $\theta = \{\theta_E, \theta_D, \theta_R, \theta_Q\}$, H is the prediction horizon, $\lambda \in (0, 1]$ is a constant that balances the contribution of each rollout step, r_t is the ground-truth reward, G_t is TD target, and $\hat{z}_{t+1}^i = D^i\left(z_t^i, a_t^i, e_t^i\right)$ is the predicted latent state. The loss can be backpropagated to the encoder via $z_t^i = E^i\left(o_t^i\right)$, so do the dynamics and reward losses. The latent target $z_{t+1}^i = E^i\left(o_{t+1}^i\right)$ is detached with the stop-gradient operator $\text{sg}(\cdot)$ to prevent cyclic gradient flow. Soft Cross-Entropy loss is used to match the discretized reward and Q-value predictions, with details provided in Appendix B.3. Additionally, to ensure modularity and scalability, each agent's world model is trained independently, and the loss can not be backpropagated through the communication channel.

Based on Eq. (3), the encoder learns a compact latent space, while the dynamics and reward predictors minimize prediction errors in this space, ensuring alignment with real environment dynamics. The actor generates initial action estimates in the latent space to warm-start planning and is trained using the Heterogeneous-Agent Soft Actor-Critic (HASAC) (Liu et al., 2024a) algorithm:

$$\mathcal{L}(\psi) = \sum_t^H \lambda^t \left(Q^i\left(z_t^i, \hat{a}_t^i, e_t^i\right) - \alpha \mathcal{H}\left[\pi^{i,\text{Act}}\left(\cdot \left| z_t^i, e_t^i\right.\right)\right] \right), \tag{4}$$

where α is the entropy coefficient, and $\mathcal{H}[\cdot]$ denotes the entropy function.

Sequential Update Scheme. We adopt the sequential update scheme (Kuba et al., 2022; Zhong et al., 2024) to train world models in a manner aligned with its autoregressive structure. When training the agent v^{i+1}, its inputs are conditioned on the predictions of the first i agents, produced by their most recently updated models. This preserves the sequential dependency, ensuring predictions exploit the most up-to-date outputs, which stabilizes training and improves monotonicity across agent indices.

Random Masking. Inspired by Masked AutoEncoders (He et al., 2022) and their applications in MARL (Kang et al., 2025), we randomly permute the communication order among agents and allow each agent to skip communication with a certain probability. This random masking simulates realistic communication interruptions and forces the world model to robustly adapt to communication uncertainties, significantly enhancing the model's resilience against communication failures.

4.2 PLANNING WITH SEQUENTIAL COMMUNICATION

Although Eq. (2) includes an actor, it does not serve directly as an explicit decision policy; instead, it provides initial action estimates for the planner. We next propose a sequential multi-agent planner based on Model Predictive Path Integral (MPPI) (Williams et al., 2015) that leverages the predictions of world models to optimize each agent's action. In this framework, the actor contributes by narrowing the action search space to promising regions, while the planner ensures robust long-term decision-making and corrects suboptimal proposals from the actor.

At each timestep t, agent v^i samples N candidate action sequences of horizon H, denoted $a_{t:t+H}^i$, from the initial distribution guided by the actor. Conditioned on its latent state and the received message, the agent performs latent rollouts with its local world model to predict future trajectories.

---

[1] We implement the concatenation using a mask-based scheme to ensure consistent dimensionality, with details provided in Appendix A.1.

The value of each trajectory is then estimated as

$$V_{t+H}^i = \gamma^H Q^i(\hat{z}_{t+H}^i, a_{t+H}^i, e_{t+H}^i) + \sum_{h=t}^{t+H-1} \gamma^{h-t} R^i(\hat{z}_h^i, a_h^i, e_h^i), \qquad (5)$$

where V_{t+H}^i represents the value estimate of a sampled action sequence, computed as the sum of predicted rewards over the horizon plus the terminal value given by the critic. Then, candidate sequences are ranked according to their evaluated values, and the highest-scoring subset is selected as the elite set. The action distribution is updated toward the statistics of these elite trajectories, thereby concentrating future sampling around high-value regions and progressively refining the search space. [2]

After several iterations until convergence, the optimized action sequence and predicted trajectory are transmitted as a communication message to the next agent, which repeats the same planning procedure. This sequential communication–planning paradigm substantially reduces communication overhead and enhances multi-agent cooperation efficiency through explicit intention sharing.
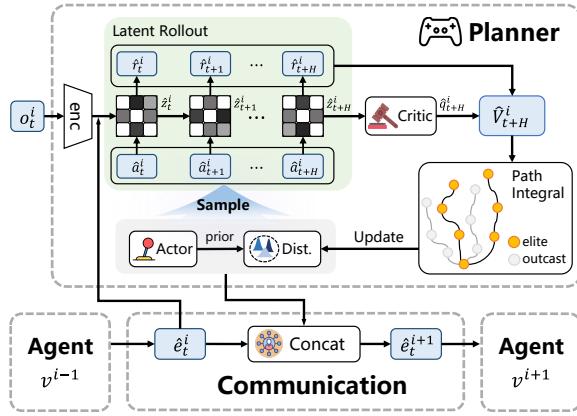

Figure 2: Sequential multi-agent planner: each agent optimizes its action sequence via local world model rollouts and critic evaluations, then passes the optimized trajectory to the next agent for efficient sequential cooperation.

Low-pass Action Smoothing. To prevent mechanical wear caused by abrupt action changes, we integrate a low-pass filtering strategy (Kicki, 2025). In each planning iteration, sampled action sequences are filtered along the temporal dimension to suppress high-frequency fluctuations. This smoothing enforces gradual action transitions across timesteps, reducing control discontinuities and promoting stable, consistent behavior on physical robots. Further details are provided in Appendix A.2.

Heuristic Early-Stopping. Considering the computational constraints of physical robotic platforms, we design a motion-planning heuristic that terminates iterations when the KL (Kullback & Leibler, 1951) divergence between consecutive action distributions falls below a threshold. This early-stopping criterion mitigates diminishing returns (Kobilarov, 2012), reducing computation while preserving plan quality. Further details and experiments regarding this design are provided in Appendix C.5.

Communication Cache. Inspired by Q-chunking (Li et al., 2025) which reuses temporally extended action units to improve decision efficiency, we introduce a communication cache that stores the predicted messages from the previous agent, enabling the current agent to retrieve them when communication fails. For instance, if communication fails at $t + 1$, agent v^{i+1} retrieves the cached message $z_{t+1}^i = D^i(E^i(o_t^i))$ from agent v^i instead of the ideally updated message $\hat{z}_{t+1}^i = E^i(o_{t+1}^i)$.

5 EXPERIMENTS

Environments. We evaluate SeqWM and baselines in two challenging multi-robot cooperative environments: Bimanual Dexterous Hands (Bi-DexHands) (Chen et al., 2024a) and Multi-Quadruped Environment (Multi-Quad) (Xiong et al., 2024). In Bi-DexHands, two agents control a pair of dexterous hands to accomplish high-dimensional manipulation tasks (up to $\mathcal{O} \in \mathbb{R}^{229}, \mathcal{A} \in \mathbb{R}^{26}$). In Multi-Quad, multiple quadruped robots collaborate to solve coordination tasks, and we further deploy SeqWM on real Unitree Go2-W robots to assess its sim-to-real transfer.

5.1 COMPARISONS

Baselines. We select several competitive baselines, including: HASAC (Liu et al., 2024a), a state-of-the-art model-free method extending SAC to multi-agent settings; MARIE (Zhang et al., 2025a), a

---

[2] The details of the planner and its implementationare provided in Appendix A.1 and B.

model-based method employing a Transformer for dynamics prediction; MAT (Wen et al., 2022), a method adopting the sequential decision-making paradigm; and MAPPO (Yu et al., 2022), a most widely used algorithm, included as a general-purpose baseline.
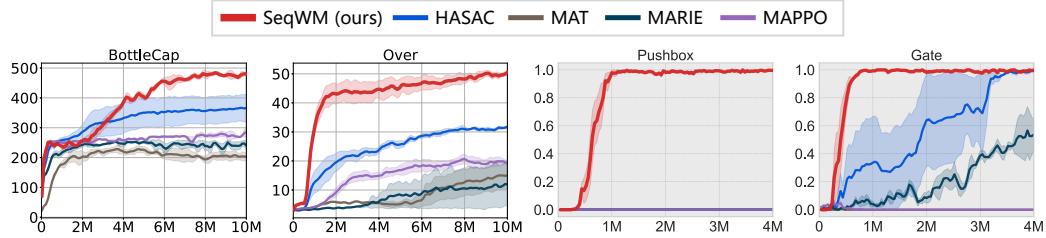

Figure 3: Performance comparisons on selected tasks of SeqWM with other baselines. Task in Bi-DexHands report the episode return, while Multi-Quad (gray background) reports success rate. Bold lines indicate the mean over multiple seeds, with shaded regions denoting the 95% confidence intervals. The results on all other tasks are reported in Figure 13 in Appendix C.1.

Results on Bi-DexHands. The representative tasks in Bi-DexHands include object transfer tasks (`Over`,`CatchAbreast`,`CatchOver2Underarm`), which require the two hands to transfer an object under different relative positions and grasping postures; and functional manipulation tasks (`BottleCap`,`Pen`,`Scissors`), which involve precise bimanual operations to achieve specific functional goals, such as opening a bottle cap, removing a pen lid, or spreading a pair of scissors.

As shown in Figure 3, SeqWM achieves higher asymptotic returns and faster convergence across all tasks. In several tasks (`Over`, `CatchOver2Underarm`, `Scissors`), SeqWM reaches near-optimal performance within 2–4M steps, while baselines require far more interactions or fail to match it. In more challenging tasks (`Pen`, `CatchAbreast`), SeqWM steadily improves and achieves the highest final returns with lower variance, demonstrating stability. Remarkably, in `BottleCap`, SeqWM is the only method that successfully grasps the bottle body and opens the cap.

Results on Multi-Quad. In `Gate`, the robots are required to pass through a narrow gate as quickly as possible without collision. In `PushBox`, they jointly push a large box to a designated target location. In `Shepherd`,, the two quadruped robots (as sheepdogs) cooperatively guide another robot (as sheep) to a target area (as sheep pen).

In `Gate` and `Shepherd`, it rapidly approaches near-100% success rates within the early phase, significantly surpassing baselines in terms of sample efficiency. In `PushBox`, SeqWM is the only method capable of completing the task, whereas all baselines fail to push the box to the target. [3] This superior performance stems from SeqWM's sequential world model, which enables each agent to plan actions conditioned on its predecessors' intentions, thereby enhancing coordination.

5.2 EMPOWERED COOPERATIVE BEHAVIORS

We further visualize the behaviors learned by SeqWM, showing that it not only acquires stable policies in high-dimensional state and action spaces, but also achieves advanced cooperative behaviors, including **predictive adaptation**, **temporal alignment**, and **role division**.

Bi-DexHands Behaviors. In `Catch-Over2Underarm`, the throwing hand first performs prediction and planning, explicitly transmitting future trajectories to the catching hand. Guided by this message, the catching hand then exhibits **predictive adaptation** by anticipating the object's motion and landing point and proactively adjusting its grasping posture. As shown in Frames C–D, the catching hand lowers and opens in advance, aligning its posture with the predicted landing point to enable a reliable grasp. In `Pen`, two hands achieve near-perfect **temporal alignment** by exchanging predictions of future actions in advance. As a result, they grasp the pen body and cap almost simultaneously in Frame D and efficiently complete the extraction in Frames E–F, substantially enhancing cooperative efficiency.

Multi-Quad Behaviors. Figure 5 further shows the role division learned by SeqWM in the Multi-Quad-`PushBox`. In Frames A–B ($t = 1 \rightarrow 2$), the two quadruped robots navigate to opposite sides

---

[3] SeqWM-MARL shows the behavior comparison between SeqWM and the next-best baseline on `BottleCap` and `PushBox`.
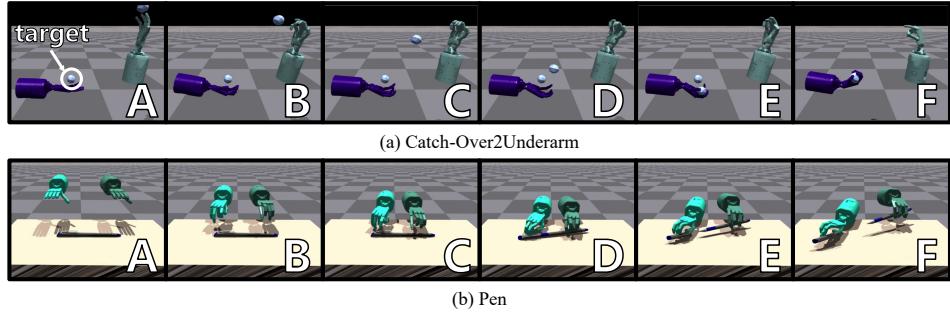
(a) Catch-Over2Underarm

(b) Pen

Figure 4: Trajectory visualizations of `Catch-Over2Underarm` and `Pen` with SeqWM.

of the box, establishing an effective pushing configuration. In Frames B–D ($t = 2 \rightarrow 4$),, both maintain high positive x-axis velocities, indicating continuous forward pushing force. At Frame C ($t = 3$), Robot 2 produces a downward y-axis velocity, adjusting the push direction toward the target, while Robot 1 gradually increases its negative y-axis velocity to assist in directional control. As the box approaches the target, Robot 1 reduces its x-axis velocity to avoid overshooting. These behaviors demonstrate that SeqWM supports not only effective force coordination but also fine-grained directional adjustments, resulting in precise and efficient task completion.

Figure 5: Behavior visualizations in `PushBox`. The first row shows the execution process, where the box is significantly larger than the robots, requiring coordinated efforts from both quadrupeds to complete the task. The left side of second row visualizes the trajectories of the robots and the box, with the right side showing the x-axis and y-axis velocities and orientations of each robot.

5.3 SCALABILITY TO MORE AGENTS

We extend the `Gate` to 3–5 agents to evaluate the scalability of SeqWM with respect to the number of agents, and the behavioral visualizations of `5-robot-Gate` are presented in Figure 6.
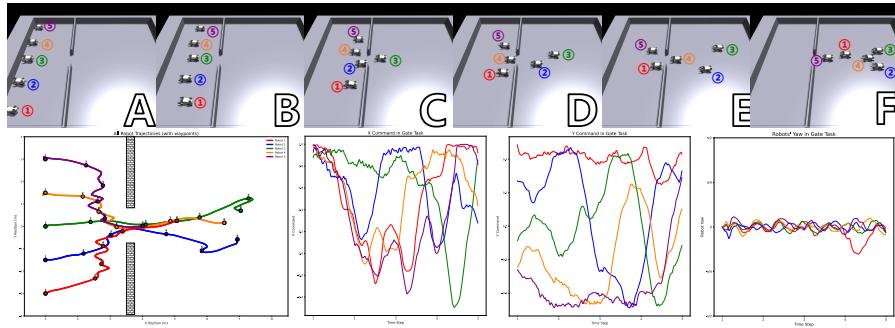

Figure 6: Visualization of the learned behaviors on `5-robot-Gate`.

As the robots approach the narrow gate, they exhibit predictive adaptation, with certain agents proactively decelerating or waiting to avoid potential congestion. For instance, at Frame B ($t \approx 2$), Robot 3 maintains a near-unity positive x-command, while the other robots moderately reduce

their forward commands. In terms of temporal alignment, the x-command trajectories reveal a clear wave-like alternation, where the peak sequence *(3 → 2 → 1 → 4 → 5)* mirrors the actual passing order, reflecting a dynamic "first-pass–then-follow" sequence. Overall, the team establishes a coordinated rhythm of "prediction–waiting–passing–yielding," which enables efficient multi-robot traversal under constrained environmental conditions. Further quantitative analyses and visualizations on `3-5 robot Gate` and `Mujoco-6a-Cheetah` are provided in Appendix C.2.

5.4 REAL-WORLD DEPLOYMENT

The real-world experimental setup is detailed in Appendix C.6, and the results are shown in Figure 7.
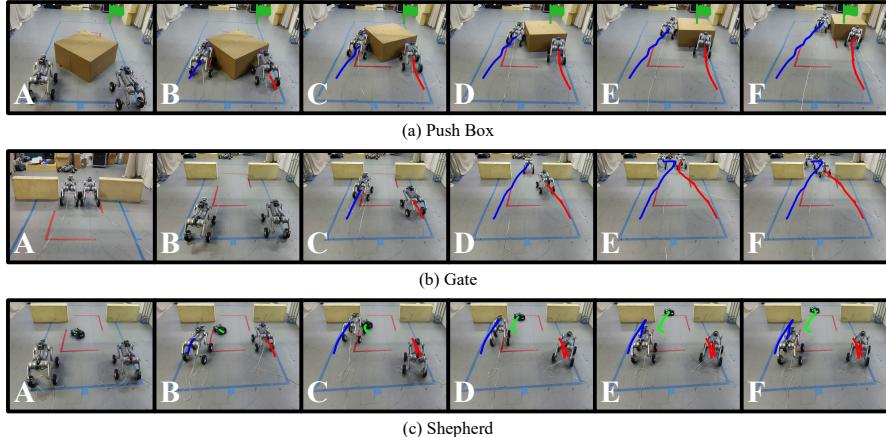

(a) Push Box

(b) Gate

(c) Shepherd

Figure 7: Real-world results of multi-robot cooperation tasks. The trajectories of Robot 1, Robot 2, and the Sheep are marked in different colors.

In `PushBox`, the two quadrupeds approach the box from opposite sides and coordinate their pushing forces and directions to move it toward the target. Between Frames D-F, Robot 1 moves forward to provide the main pushing force, while Robot 2 makes slight lateral adjustments to steer the box. The overall pushing pattern, including the division of roles and the gradual directional adjustments, closely matches the behavior observed in simulation, confirming a successful sim-to-real transfer.

In `Gate`, two clear yielding events are observed. Between Frames C-D, Robot 1 slows down and waits for Robot 2 to pass first, demonstrating priority management in constrained spaces. After crossing Frames E-F, Robot 2 actively veers aside to leave sufficient space for Robot 1, enabling smooth passage without collisions. These behaviors reflect SeqWM's trajectory prediction and intention-sharing capabilities, allowing natural, efficient yielding without high-frequency communication.

In `Shepherd`, Robot 1 accelerates between Frames A-B, causing the Sheep to move left. To prevent the Sheep from hitting the left gate frame, Robot 1 retreats while Robot 2 advances between Frames C-D. This maneuver drives the Sheep away from Robot 2 and into the target area. The sequence highlights SeqWM's capacity for predictive coordination and adaptive role allocation, where the one agent's motion influences the sheep robot's response and the another agent adapts accordingly to achieve the common goal.

5.5 ABLATION STUDIES

Sequential Sample Generation. To evaluate the contribution of the sequential paradigm in SeqWM's world model, we replace it with centralized and decentralized architectures, ensuring all models have an equal number of parameters for a fair comparison. Using `BottleCap`, we collect 50K environment steps with random actions and train each model for 2.5K steps using the loss in Eq. (3). After training, we gather 1K additional steps to measure dynamics and reward prediction errors across different horizons. As shown in Figure 8, the sequential and centralized models achieve similarly low errors, both substantially outperforming the decentralized model. The results confirm the advantage of sequential prediction, where each agent conditions its output on the predictions of its predecessors, yielding more accurate and coherent rollouts.
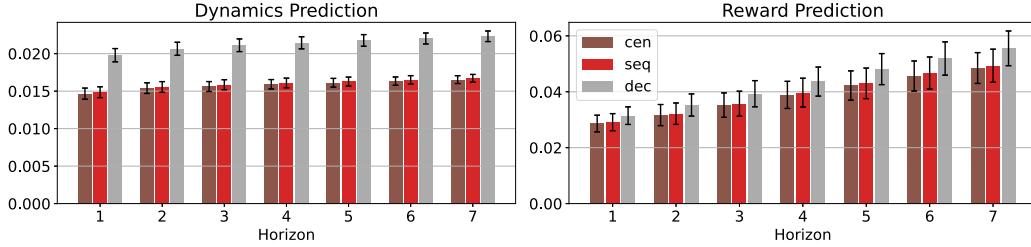
Figure 8: Accuracy of the world model prediction.

Communication Function. We replaced the concat communication function in SeqWM with alternative fusion mechanisms, including MLP, cross-attn, and RNN, and evaluated them on `BottleCap`. The results in Figure 9 show that the simplest concat approach achieves the highest and most stable performance. This advantage stems from two factors: (i) concat preserves the complete communication content, allowing the dynamics and reward predictors to autonomously identify and exploit the most informative features during training; and (ii) it introduces no additional learnable parameters, thereby maintaining stable gradient propagation in long-horizon prediction. Moreover, we observe that RNN-based fusion even underperforms the no-communication baseline (dec), which we attribute to its sensitivity to input ordering—an undesirable property in multi-agent communication scenarios lacking a fixed semantic sequence.

Figure 9: Ablation study of the communication function in SeqWM.

5.6 ADDITIONAL EXPERIMENTS

To further validate the effectiveness of SeqWM, we present the following additional analyses:

1) Visualizations of the learned behaviors on all other tasks in Appendix C.3 and SeqWM-MARL;
2) An ablation analysis of the complementary roles of the actor and planner in Appendix C.4;
3) An evaluation of SeqWM's scalability with respect to the number of agents in Appendix C.2;
4) An experimental analysis of time consumption and early stopping in the planner in Appendix C.5.

6 CONCLUSION

This paper presented SeqWM, a novel model-based MARL framework that integrates the sequential paradigm into world model learning and planning. By structurally decomposing joint dynamics into autoregressive, agent-wise models, SeqWM offers a principled approach that reduces modeling complexity and naturally enables intention sharing through predicted trajectories. This methodological innovation not only improves scalability but also facilitates the emergence of advanced cooperative behaviors such as predictive adaptation, temporal alignment, and role division. Extensive experiments in Bi-DexHands and Multi-Quad show that SeqWM achieves state-of-the-art performance with superior sample efficiency, while real-world deployment on quadruped robots confirms that these cooperative behaviors transfer reliably from simulation to physical platforms. Beyond empirical results, SeqWM demonstrates that sequential paradigms provide an efficient and scalable principle for structuring multi-agent cooperation, paving the way for more robust and efficient deployment of cooperation in physical multi-robot systems.

Future Work. Benefiting from the integration of the sequential paradigm and agent-wise world models, SeqWM naturally extends to **heterogeneous robot teams** and **human–robot semantic understanding**. With each agent maintaining an independent world model, the framework accommodates diverse dynamics and sensing modalities, enabling cooperation among quadrupeds, manipulators, and aerial robots. Moreover, the explicit trajectory rollouts can be shared not only across robots but also with humans as interpretable intention signals, fostering transparent collaboration, mutual understanding, and trust in human–robot teams.
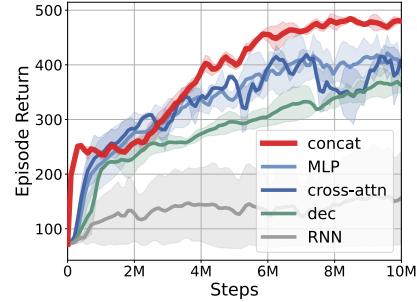
ETHICS STATEMENT

This work focuses on advancing multi-robot cooperation through model-based reinforcement learning. All experiments were conducted in simulated environments and on standard quadruped robot platforms, with no involvement of humans, animals, or sensitive personal data. The proposed methodology focuses on technical contributions to the fields of reinforcement learning, multi-agent systems, and robotics, without ethical implications beyond standard academic research practices.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of the results, we provide open-source code at `https://github.com/zhaozijie2022/seqwm-marl`. Hyperparameters and training procedures are detailed in Appendix B. All baseline comparisons use publicly available implementations, with the documented parameter settings as referenced in the respective sections.

USE OF LARGE LANGUAGE MODELS

Large Language Models (LLMs) were used in the preparation of this paper exclusively for writing assistance and language polishing. The conceptualization of the research, methodology design, experimental implementation, and analysis were all conducted entirely by the authors. The authors take full responsibility for the content of this paper.

REFERENCES

Michel Aractingi, Pierre-Alexandre Léziart, Thomas Flayols, Julien Perez, Tomi Silander, and Philippe Souères. Controlling the solo12 quadruped robot with deep reinforcement learning. *scientific Reports*, 13(1):11945, 2023. URL `https://dl.acm.org/doi/abs/10.5555/3600270.3601471`.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 449–458. PMLR, 06–11 Aug 2017. URL `https://proceedings.mlr.press/v70/bellemare17a.html`.

Jiajun Chai, Yuqian Fu, Dongbin Zhao, and Yuanheng Zhu. Aligning credit for multi-agent cooperation via model-based counterfactual imagination. AAMAS '24, pp. 281–289, 2024. URL `https://dl.acm.org/doi/abs/10.5555/3635637.3662876`.

Yuanpei Chen, Yiran Geng, Fangwei Zhong, Jiaming Ji, Jiechuang Jiang, Zongqing Lu, Hao Dong, and Yaodong Yang. Bi-dexhands: Towards human-level bimanual dexterous manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):2804–2818, 2024a. URL `https://ieeexplore.ieee.org/abstract/document/10343126`.

Zixuan Chen, Xialin He, Yen-Jen Wang, Qiayuan Liao, Yanjie Ze, Zhongyu Li, S. Shankar Sastry, Jiajun Wu, Koushil Sreenath, Saurabh Gupta, and Xue Bin Peng. Learning smooth humanoid locomotion through lipschitz-constrained policies. 2024b. URL `https://arxiv.org/abs/2410.11825`.

Guilherme Christmann, Ying-Sheng Luo, Hanjaya Mandala, and Wei-Chao Chen. Benchmarking smoothness and reducing high-frequency oscillations in continuous control policies. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 627–634, 2024. URL `https://ieeexplore.ieee.org/abstract/document/10802057`.

Gang Ding, Zeyuan Liu, Zhirui Fang, Kefan Su, Liwen Zhu, and Zongqing Lu. Multi-agent coordination via multi-level communication. *Advances in Neural Information Processing Systems*, 37:118513–118539, 2024. URL `https://openreview.net/forum?id=3l2HnZXNou`.

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, AAMAS '22, pp. 381–390, Richland, SC, 2022. ISBN 9781450392136. URL https://dl.acm.org/doi/abs/10.5555/3535850.3535894.

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

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2022. URL https://ieeexplore.ieee.org/abstract/document/9879206.

Kun Hu, Muning Wen, Xihuai Wang, Shao Zhang, Yiwei Shi, Minne Li, Minglong Li, and Ying Wen. Pmat: Optimizing action generation order in multi-agent reinforcement learning. AAMAS '25, pp. 997–1005, 2025. URL https://dl.acm.org/doi/abs/10.5555/3709347.3743619.

Sehyeok Kang, Yongsik Lee, Gahee Kim, Song Chong, and Se-Young Yun. Ma2e: Addressing partial observability in multi-agent reinforcement learning with masked auto-encoder. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=klpdEThT8q.

Nouman Khan. *Sequential Decision Making in Cooperative Multi-Agent Systems with Constraints*. PhD thesis, University of Michigan, 2025. URL https://deepblue.lib.umich.edu/handle/2027.42/199212. Accessed: 2025-09-08.

Piotr Kicki. Low-pass sampling in model predictive path integral control. 2025. URL https://arxiv.org/abs/2503.11717.

Marin Kobilarov. Cross-entropy motion planning. *The International Journal of Robotics Research*, 31(7):855–871, 2012. URL https://doi.org/10.1177/0278364912444543.

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.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951. URL https://www.jstor.org/stable/2236703.

Patrick Lancaster, Nicklas Hansen, Aravind Rajeswaran, and Vikash Kumar. Modem-v2: Visuo-motor world models for real-world robot manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7530–7537, 2024. URL https://ieeexplore.ieee.org/abstract/document/10611121.

Samuel Lavoie, Christos Tsirigotis, Max Schwarzer, Ankit Vani, Michael Noukhovitch, Kenji Kawaguchi, and Aaron Courville. Simplicial embeddings in self-supervised learning and downstream classification. 2022. URL https://arxiv.org/abs/2204.00616.

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. *arXiv preprint arXiv:2507.07969*, 2025.

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*, 2024a. URL https://openreview.net/forum?id=tmqOhBC4a5.

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*, 2024b. URL https://openreview.net/forum?id=CpnKq3UJwp.

Diganta Misra. Mish: A self regularized non-monotonic activation function. 2020. URL https://arxiv.org/abs/1908.08681.

R.R. Negenborn and J.M. Maestre. Distributed model predictive control: An overview and roadmap of future research opportunities. *IEEE Control Systems Magazine*, 34(4):87–97, 2014. URL https://ieeexplore.ieee.org/abstract/document/6853439.

Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016. URL https://link.springer.com/book/10.1007/978-3-319-28929-8.

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.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. volume 155 of *Proceedings of Machine Learning Research*, pp. 1049–1065. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/pinneri21a.html.

Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Strengthening generative robot policies through predictive world modeling. 2025. URL https://arxiv.org/abs/2502.00622.

Xujie Song, Liangfa Chen, Tong Liu, Wenxuan Wang, Yinuo Wang, Shentao Qin, Yinsong Ma, Jingliang Duan, and Shengbo Eben Li. Lipsnet++: Unifying filter and controller into a policy network. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=KZo2XhcSg6.

Jiankai Sun, Yiqi Jiang, Jianing Qiu, Parth Nobel, Mykel J Kochenderfer, and Mac Schwager. Conformal prediction for uncertainty-aware planning with diffusion dynamics model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 80324–80337. Curran Associates, Inc., 2023. URL https://dl.acm.org/doi/abs/10.5555/3666122.3669643.

Edan Toledo. Codreamer: Communication-based decentralised world models. In *Coordination and Cooperation for Multi-Agent Reinforcement Learning Methods Workshop*, 2024. URL https://openreview.net/forum?id=f2bgGy7Af7.

Bogdan Vlahov, Jason Gibson, David D Fan, Patrick Spieler, Ali-akbar Agha-mohammadi, and Evangelos A Theodorou. Low frequency sampling in model predictive path integral control. *IEEE Robotics and Automation Letters*, 9(5):4543–4550, 2024. URL https://ieeexplore.ieee.org/abstract/document/10480553.

Xihuai Wang, Zheng Tian, Ziyu Wan, Ying Wen, Jun Wang, and Weinan Zhang. Order matters: Agent-by-agent policy optimization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Q-neeWNVv1.

Yinuo Wang, Wenxuan Wang, Xujie Song, Tong Liu, Yuming Yin, Liangfa Chen, Likun Wang, Jingliang Duan, and Shengbo Eben Li. ODE-based smoothing neural network for reinforcement learning tasks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=S5Yo6w3n3f.

Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. *Advances in Neural Information Processing Systems*, 35:16509–16521, 2022. URL https://openreview.net/forum?id=1W8UwXAQubL.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. 2015. URL https://arxiv.org/abs/1509.01149.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. volume 205 of *Proceedings of Machine Learning Research*, pp. 2226–2240. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/wu23c.html.

Ziyan Xiong, Bo Chen, Shiyu Huang, Wei-Wei Tu, Zhaofeng He, and Yang Gao. Mqe: Unleashing the power of interaction with multi-agent quadruped environment. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5918–5924. IEEE, 2024. URL https://ieeexplore.ieee.org/abstract/document/10801682.

Kaixuan Xu, Jiajun Chai, Sicheng Li, Yuqian Fu, Yuanheng Zhu, and Dongbin Zhao. DipLLM: Fine-tuning LLM for strategic decision-making in diplomacy. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=hfPaOxDWfI.

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. *Advances in neural information processing systems*, 35:24611–24624, 2022. URL https://openreview.net/forum?id=YVXaxB6L2Pl.

Chao Yu, Xinyi Yang, Jiaxuan Gao, Jiayu Chen, Yunfei Li, Jijia Liu, Yunfei Xiang, Ruixin Huang, Huazhong Yang, Yi Wu, and Yu Wang. Asynchronous multi-agent reinforcement learning for efficient real-time multi-robot cooperative exploration. AAMAS '23, pp. 1107–1115, 2023. URL https://dl.acm.org/doi/abs/10.5555/3545946.3598752.

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*, 2025a. URL https://openreview.net/forum?id=xT8BEgXmVc.

Yang Zhang, Xinran Li, Jianing Ye, Delin Qu, Shuang Qiu, Chongjie Zhang, Xiu Li, and Chenjia Bai. Revisiting multi-agent world modeling from a diffusion-inspired perspective. 2025b. URL https://arxiv.org/abs/2505.20922.

Zijie Zhao, Yuqian Fu, Jiajun Chai, Yuanheng Zhu, and Dongbin Zhao. Meta learning task representation in multiagent reinforcement learning: From global inference to local inference. *IEEE Transactions on Neural Networks and Learning Systems*, 36(8):14908–14921, 2025a. URL https://ieeexplore.ieee.org/abstract/document/10905042.

Zijie Zhao, Zhongyue Zhao, Kaixuan Xu, Yuqian Fu, Jiajun Chai, Yuanheng Zhu, and Dongbin Zhao. Learning and planning multi-agent tasks via a moe-based world model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=fi24ry0BX5.

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.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61885–61896. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhou24f.html.

Xiaomeng Zhu, Yuyang Li, Leiyao Cui, Pengfei Li, Huan ang Gao, Yixin Zhu, and Hao Zhao. Afford-x: Generalizable and slim affordance reasoning for task-oriented manipulation. 2025a. URL https://arxiv.org/abs/2503.03556.

Xiaomeng Zhu, Changwei Wang, Haozhe Wang, Xinyu Liu, and Fangzhen Lin. Ootsm: A decoupled linguistic framework for effective scene graph anticipation. 2025b. URL https://arxiv.org/abs/2509.05661.

APPENDICES

A   MULTI-AGENT PLANNER

A.1   PLANNING PROCESS

As shown in Figure 10, at timestep t, the action planning process for agent v^i can be divided into the following steps:

S1 - Communication. Agents are organized to exchange messages in a sequential manner. Specifically, agent v^i receives a message e_t^i that aggregates the predicted latent states and planned actions from all its predecessors:

$$e_t^i = \begin{cases} \emptyset, & i = 1, \\ \bigoplus_{j<i} \left(\hat{z}_t^j, a_t^j \right), & i > 1, \end{cases} \quad (6)$$

where $\oplus$ denotes concatenation. To implement this efficiently, we employ a masking-based concatenation scheme: a

Figure 10: The multi-agent planner in SeqWM.

fixed-length vector of dimension $n \times (|\mathcal{A}| + d_z)$ is pre-allocated, where n is the number of agents, $|\mathcal{A}|$ and d_z are the action and latent dimensions. Agent v^1 maintains an empty message, while subsequent agents sequentially fill in their designated slots with their own predictions $(\hat{z}_t^i, a_t^i)$ in addition to forwarding the received content. This design ensures that information is progressively accumulated along the communication chain with linear complexity in the number of agents.

S2 - Action Sampling. The planner samples N candidate action sequences from two sources. We sample N_p candidate action sequences from a diagonal Gaussian distribution $a_{t:t+H}^i \sim \mathcal{N}\left(\mu_{t:t+H}^i, (\sigma_{t:t+H}^i)^2 I\right)$, where $\mu_{t:t+H}^i, \sigma_{t:t+H}^i \in \mathbb{R}^{|\mathcal{A}| \times H}$ represent the mean and standard deviation of the H-step horizon actions. Additionally, we sample N_a action sequences directly from the actor module $\hat{a}_h^i \sim \pi^{i,\text{Act}}(\cdot|o_h^i, e_h^i), h = t : t + H$, and combine these two sets of action sequences to form N candidate action sequences.

S3 - World Model Prediction. Following sampling, the world model predicts H-step trajectories for each sampled action sequence using Eq. (2), generating N predicted sequences $\Gamma = \{(\hat{z}_h^i, a_h^i, \hat{r}_h^i)\}_{h=t:t+H}$.

S4 - Value Evaluation. Each predicted trajectory is assigned a value via the H-step return, combining the short-term cumulative predicted reward with the terminal value from the critic:

$$V_\Gamma^i = \gamma^H Q^i(\hat{z}_{t+H}^i, a_{t+H}^i, e_{t+H}^i) + \sum_{h=t}^{t+H-1} \gamma^{h-t} \hat{r}_h^i. \quad (7)$$

S5 - Action Optimization. The candidate action sequences are ranked by their evaluated values, and the top M are chosen as the elite set Γ^*. The parameters of the action distribution are updated based on the elite set using:

$$\mu_{t:t+H}^{i,(k+1)} = \frac{\sum_{m=1}^M \alpha_m \Gamma_m^*}{\sum_{m=1}^M \alpha_m}, \quad \sigma_{t:t+H}^{i,(k+1)} = \sqrt{\frac{\sum_{m=1}^M \alpha_m \left(\Gamma_m^* - \mu_{t:t+H}^{i,(k+1)}\right)^2}{\sum_{m=1}^M \alpha_m}}, \quad (8)$$

where the weights are generated based on the evaluated values as $\alpha_m = \exp\left[\tau\left(V_{\Gamma_m^*} - \max_{m\in M} V_{\Gamma_m^*}\right)\right]$, with τ being the temperature coefficient.

Iteration. For the default setting, the above process are iterated K times to derive the final action distribution. If the early-stopping heuristic is applied, after each iteration, we check whether the action optimization has converged by evaluating the KL divergence between the current and previous action distributions, $\mathbb{D}_{KL}(\mathcal{N}^{(k+1)} \| \mathcal{N}^{(k)}) < \eta$, where η is a small threshold.

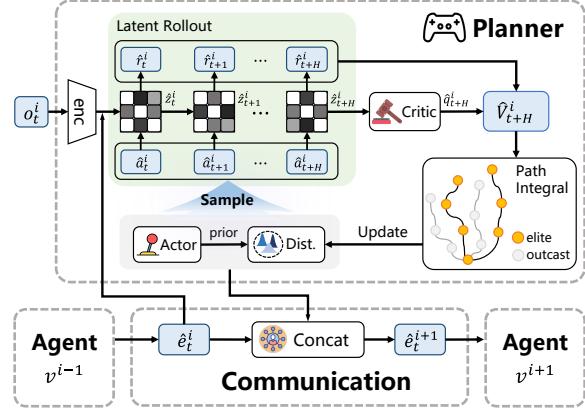The detailed hyperparameters used in the model-based planner are summarized in Table 2.

A.2 Low-Pass Action Smoothing

In real-world robotics, high-frequency changes in control inputs can cause severe mechanical impacts, accelerating wear and reducing execution stability. Therefore, many studies in reinforcement learning and motion planning incorporate action-smoothing constraints, such as adding penalties on differences between consecutive actions during policy updates (Aractingi et al., 2023; Christmann et al., 2024; Wang et al., 2025), introducing regularization in policy networks (Chen et al., 2024b; Song et al., 2025), or filtering noise in trajectory optimization (Pinneri et al., 2021; Vlahov et al., 2024; Kicki, 2025), to reduce jitter and improve executability. Inspired by these methods, we apply frequency-domain low-pass filtering directly to the sampled action noise in our planner, explicitly suppressing the high-frequency components of the actions.
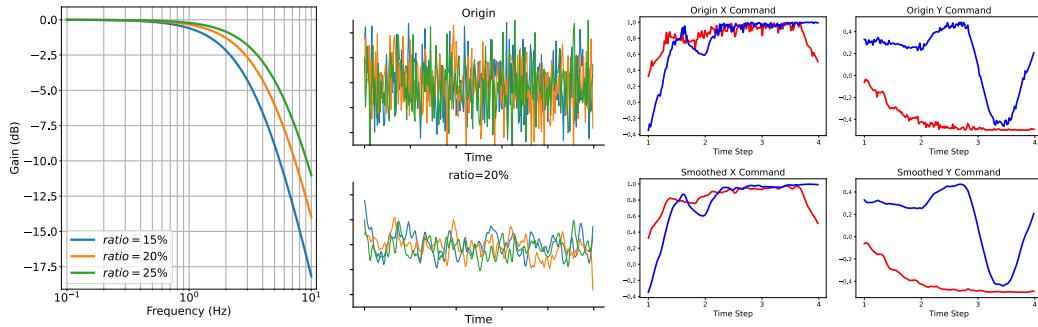

Figure 11: Visualization of the low-pass filter. (*Left*): Amplitude–frequency response of the low-pass filter. (*Middle*): Filtering effects on random signals at 20% cutoff ratios, the different colors represent different action dimensions. (*Right*): Effects of low-pass filtering on control commands in `PushBox`, with different colors representing different agents.

Specifically, during action sampling, we first sample noise from a standard normal distribution, apply low-pass filtering, and then add the filtered noise to the action mean to generate candidate action sequences. We use a Butterworth filter with $o_{\mathrm{LBF}} = 1$, whose transfer function and amplitude–frequency response are given by:

$$H(s) = \frac{2\pi f_c}{s + 2\pi f_c}, \quad |H(f)| = \frac{f_c}{\sqrt{f^2 + f_c^2}}, \tag{9}$$

where f_c is the low-pass cutoff frequency. As show in Figure 11, the amplitude–frequency response shows that the high-frequency components are exponentially attenuated (approaching linear decay in logarithmic coordinates). The corresponding discrete-time difference equation, obtained via bilinear transformation, is

$$y[t] = \frac{1 - \beta}{2}(x[t] + x[t-1]) - \beta y[t-1], \quad \beta = \frac{1 - \tan\left(\pi f_c / f_s\right)}{1 + \tan\left(\pi f_c / f_s\right)}, \tag{10}$$

where f_s is the sampling frequency, i.e., the frequency of the control signal.

B Implementation Details

B.1 Pseudocode

B.2 Model Architecture

The proposed sequential world model is composed of five components: encoder, dynamics predictor, reward predictor, critic, and actor, all implemented using MLPs. We report the network configurations and the number of parameters in each module on `Dex-BottleCap` ($|\mathcal{O}| = 221, |\mathcal{A}| = 26$) in Table 1.

We employ the Mish (Misra, 2020) activation function in the hidden layers, which is smooth and non-monotonic, ensuring stable gradient flow. For latent space construction, we adopt SEM Norm (Simplicial Embeddings Normalization) (Lavoie et al., 2022) to normalize the outputs of the encoder

---

Algorithm 1 Model Training
___

Input: replay buffer $\mathcal{B}$, parameterized networks $\theta_E, \theta_D, \theta_R, \theta_Q$, and ψ for encoder, dynamics predictor, reward predictor, critic, and actor, respectively;
for episode $= 1, 2, 3, \ldots,$ **do**
    for step $t = 1, 2, 3, \ldots$ **do**
        Get real data $([o_t^i]_{i=1:n}, [a_t^i]_{i=1:n}, r_t, [o_{t+1}^i]_{i=1:n})$ by interacting with the environment
        Add transition into buffer: $\mathcal{B} = \mathcal{B} \cup ([o_t^i]_{i=1:n}, [a_t^i]_{i=1:n}, r_t, [o_{t+1}^i]_{i=1:n})$
    end for
    for epoch $= 1, 2, 3, \ldots,$ **do**
        Sample trajectories from $\mathcal{B}$
        Update $\theta_E, \theta_D, \theta_R, \theta_Q$ by minimizing Eq. (3)
        Update ψ by minimizing Eq. (4).
    end for
end for

Algorithm 2 Model Planning
___

Input: learned parameters $\theta_E, \theta_D, \theta_R, \theta_Q, \psi$, hyperparameters H, K, τ, N_p, M, N_a, initial distribution;
for step $t = 1, 2, 3, \ldots$ **do**
    for agent $i = 1, 2, \ldots, n$ **do**
        Get environment observation o_t^i and encode it to latent space: $z_t^i = E^i(o_t^i)$
        if $i > 1$ **then**
            Retrieve the message from the previous agent $e_t^i = \bigoplus_{j<i} \left(\hat{z}_t^j, a_t^j \right)$
        else
            set $e_t^i = \emptyset$
        end if
        for iteration $= 1, 2, 3, \ldots, K_p$ **do**
            Sample N_a actions $a_{t:t+H}^i \sim \mathcal{N} \left(\mu_{t:t+H}^i, (\sigma_{t:t+H}^i)^2 I \right)$
            Sample N_a actions from actor $\hat{a}_h^i \sim \pi^{i,\text{Act}}(\cdot | o_h^i, e_h^i), h = t : t + H$
            Get predictions by world model rollouts, $\Gamma = \{ (\hat{z}_h^i, a_h^i, \hat{r}_h^i) \}_{h=t:t+H}$
            Evaluate the trajectories by Eq. (7) and select top-M elite action sequences
            Update action distribution following Eq. (8)
        end for
    end for
end for

---

and dynamics predictor. It employs the softmax operator to project high-dimensional latent states onto a set of fixed-length simplices, thereby forming a soft sparse representation. This approach mitigates information degradation in high-dimensional latent spaces while preserving structural properties, striking a balance between expressive capacity and optimization feasibility. As shown in Figure 12 (left), the transformation of SEM Norm is defined as:

$$\text{SEM}(z) = \oplus_{i=1}^s g^{(i)}, \quad g^{(i)} = \frac{\exp(z_{i:i+L-1})}{\sum_{j=0}^{L-1} \exp(z_{i+j})}, \tag{11}$$

where L is the dimension of the simplex, which is set to $L = 8$ in our implementation, and $s = d_z/L$ is the number of segments of the simplex.

B.3 REWARD AND VALUE PREDICTION

Discrete Regression. The reward prediction and critic in SeqWM are modeled as discrete regression problems, which helps enhance robustness to extreme values. Take reward prediction for example, we discretize the reward range into B intervals and use two-hot encoding (Bellemare et al., 2017; Hafner et al., 2025) to map the scalar reward to a B-dimensional vector. The two-hot encoding assigns weights to the two adjacent bins, reducing quantization errors and providing smoother gradients (Zhu
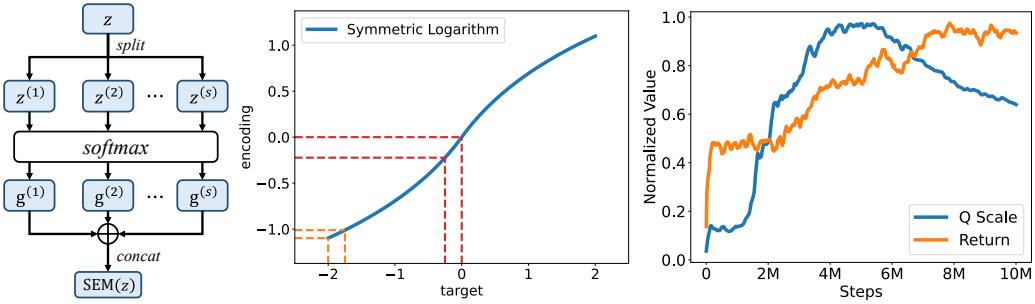
Figure 12: Overview of implementation details. (*Left*): SEM Normalization. (*Middle*): Symlog transformation, where the y-axis represents the values encoded by two-hot encoding, and the x-axis represents the values aligned with the scalar target. (*Right*): Normalized Q-scale and return in `Dex-BottleCap`.

Table 1: The network configurations.

Module	Dim	Layers	Act.	Out Act.	LayerNorm	Param.
Encoder	512	2	Mish	SEM Norm	✓	189,952
Dynamics	512	2	Mish	SEM Norm	✓	553,984
Reward	512	2	Mish	Linear	✓	342,117
Critic	256	2	Mish	Linear	✓	736,970
Actor	256	2	Mish	Tanh	✓	210,996

et al., 2025b). Formally, for a scalar reward signal r, its two-hot encoding is defined as:

$$\text{two-hot}(r)_i = \begin{cases} |b_{k+1}-r|/|b_{k+1}-b_k|, & \text{if } i = k \\ |b_k-r|/|b_{k+1}-b_k|, & \text{if } i = k+1 \\ 0, & \text{else} \end{cases}, \quad k = \sum_{j=1}^{B} \mathbb{1}(b_j \le r). \tag{12}$$

For training, we use the soft cross-entropy loss to optimize the predictor's output:

$$\text{Soft-CE}(\hat{r}, r) = -\sum_{i=1}^{B} \text{two-hot}(r)_i \times \frac{\exp(\hat{r}_i)}{\sum_j \exp(\hat{r}_j)}, \tag{13}$$

Symlog and Symexp Transformations. We employ symmetric logarithmic (symlog) and exponential (symexp) transformations (Hafner et al., 2025) to align the predicted and target values. These transformations maintain continuity and differentiability while smoothing out large numerical values, allowing the model to suppress gradient instability caused by exceptionally large values, thereby enhancing numerical stability in high-variance scenarios. These transformations are defined as:

$$\text{symlog}(x) = \text{sign}(x)\ln(|x|+1), \quad \text{symexp}(x) = \text{sign}(x)\left(\exp^{|x|}-1\right). \tag{14}$$

Figure 12 (middle) illustrates how the symlog and symexp transformations enhance numerical stability. The symlog transformation is approximately linear near $x = 0$ and logarithmic in the tails. This brings two stability benefits:

1) Sensitivity $dy/dx = 1/1+|x|$ decreases as the target value increases, leading to nearly zero gradients for extremely large values.

2) Adaptive binning for two-hot targets. For a fixed width Δx, the corresponding original scale span is $\Delta y \approx (1 + |x|)\Delta x$. Thus, extreme values are absorbed by the edge bins, preventing them from dominating the loss, while values near zero maintain fine granularity to preserve resolution.

Percentile Scaling of Critic We employ a percentile-based scaling strategy to stabilize Q-value magnitudes. Specifically, we computes the dynamic range between the 5-th and 95-th percentiles of

each batch, which effectively suppresses the influence of outliers. This range is updated smoothly using an EMA (Exponential Moving Average) controlled by a factor τ:

$$\delta^{(k+1)} = \tau\delta^{(k)} + (1 - \tau) \cdot (p_{95} - p_5),\qquad(15)$$

where p_{95} and p_5 are the 95-th and 5-th percentiles of the Q-values in the current batch, respectively, and τ is a smoothing factor (e.g., $\tau = 0.99$).

Figure 12 (right) illustrates the empirical effect of percentile scaling on Q-value magnitudes. When the return rises rapidly at the beginning of training, the scaling coefficient also increases, adapting to the broader spread of Q-values while suppressing the destabilizing impact of extreme samples. As training progresses and the return gradually converges, the dynamic range of Q-values contracts and the scaling coefficient correspondingly decreases in a smooth manner, guided by the exponential moving average. This adaptive behavior ensures that the critic remains well-conditioned across different learning phases: it expands the effective range when exploration generates diverse values, and tightens the range when the policy stabilizes. Compared to naive normalization, this percentile-based strategy provides robustness against transient spikes or rare outliers, leading to more consistent value estimation and consequently more stable planning performance.

B.4 HYPERPARAMETERS

We summarize the hyperparameters used in SeqWM in Table 2 and Table 3.

Table 2: The Notations and Values of hyperparameters in the planner.

Hyperparameters	Notations	Value	Hyperparameters	Notations	Value
rollout horizon	H	3	sampling actions	N_p	512
planning iterations	K	6	elites	M	64
temperature	τ	0.5	actor samples	N_a	24

Table 3: The hyperparameters used in the world model.

Hyperparameters	Value	Hyperparameters	Value	Hyperparameters	Value
batch size	1000	buffer size	1e6	dynamics coef	20
encoder lr scale	0.3	entropy coef	1e-4	lr	5e-4
n-step return	20	num bins	101	q coef	0.1
reward coef	0.1	step ρ	0.5		

C   ADDITIONAL EXPERIMENTS

C.1   COMPARISONS ON OTHER TASKS

We report additional comparison results on other tasks to complement Figure 3.

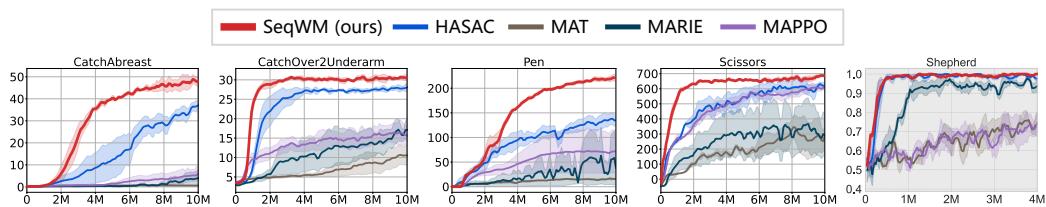Figure 13: Comparison results on other tasks.

Multi-Agent MuJoCo. To evaluate the scalability of SeqWM with respect to the number of agents, we conduct supplementary experiments on the `6a-Cheetah` in the MA-Mujoco (Multi-Agent MuJoCo) (Peng et al., 2021) environment. MA-Mujoco partitions a robot into multiple agents according to different body parts; in `6a-Cheetah`, six agents are required to coordinate to control a single joint to run faster. To achieve scalability, we modify the communication protocol in Eq. (2) to transmit only the action sequence:

$$e_t^{i+1} = e_t^i \oplus a_t^i \tag{16}$$

The results, reported in Figure 14, show that SeqWM achieves an average return exceeding 12,000 on the 6-agent Cheetah task, which is, to our best knowledge, the state-of-the-art performance for this task.

Figure 14: Performance on `6a-Cheetah`.

Figure 15: Success rates on `Gate` with more robots.

Gate with More Robots. We also evaluate scenarios with more robots on `Gate`, with success rates reported in Figure 15. As the number of robots increases, the task becomes more complex, but SeqWM still successfully learns effective cooperative strategies, achieving nearly 100% success rates in all settings within 4M time steps. Additionally, we visualize the learned behaviors of SeqWM with different numbers of robots in Figure 16, Figure 17, and Figure 18.
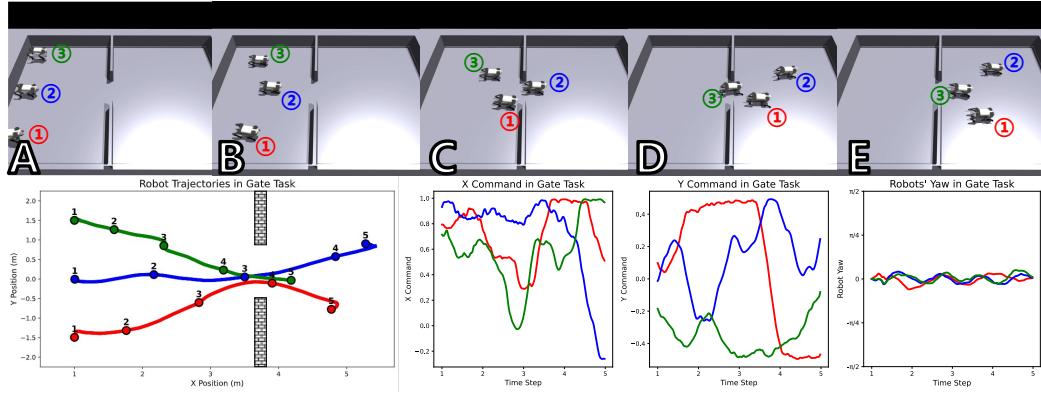

Figure 16: Visualization of the learned behaviors on `Gate` with 3 robots.

As shown in Figure 16, from $t=1$ (Frame A) to $t=2$ (Frame B) all agents head toward the passage with high x-velocity commands. Around $t \approx 2$ (Frame B), priority is established: Robot 2 maintains forward x-command while Robot 1 and Robot 3 reduce theirs, preventing congestion (see "X Command"). From $t=2 \to 3$ (Frame B $\to$ C), Robot 1 clears the gate and issues a lateral y-command to move aside and create space (panel "Y Command"), with yaw staying near zero for all agents ("Robots' Yaw"). Finally, from $t=3 \to 4$ (Frame C & D), Robot 1 passes as Robot 3 decelerates and holds position; Robot 3 proceeds last, completing a smooth, collision-free sequence *(order: 2 $\to$ 1 $\to$ 3)*.
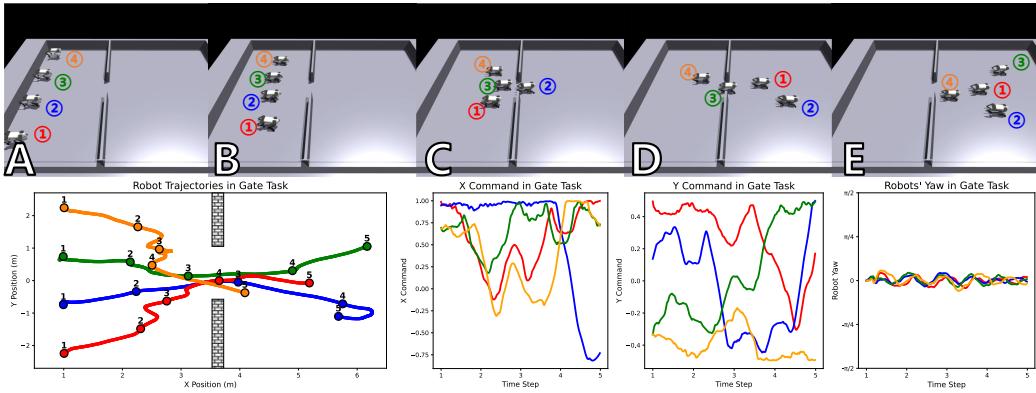
Figure 17: Visualization of the learned behaviors on `Gate` with 4 robots.

As shown in Figure 17, frames A–B show all agents accelerating toward the passage with high x-velocity commands. Around $t\approx3$ (Frame C), an implicit queue is established: Robot 2 keeps a near-unity x-command while Robot 1, Robot 3, and Robot 4 exhibit pronounced dips in the "X Command" panel, preventing blockage at the bottleneck. From $t=3\rightarrow4$ (Frame D), Robot 2 clears the gate first and issues a strong negative y-command to drift downward and vacate the corridor ("Y Command"), with yaw traces remaining close to zero for all robots ("Robots' Yaw"). Robot 1 then increases its x-command and passes second while slightly steering upward in y to avoid interference. Finally, from $t=4\rightarrow5$ (Frame E), Robot 3 proceeds third, followed by Robot 4, which had maintained the lowest x-command during the queuing phase, completing a smooth, collision-free sequence (*order: 2 → 1 → 3 → 4*).
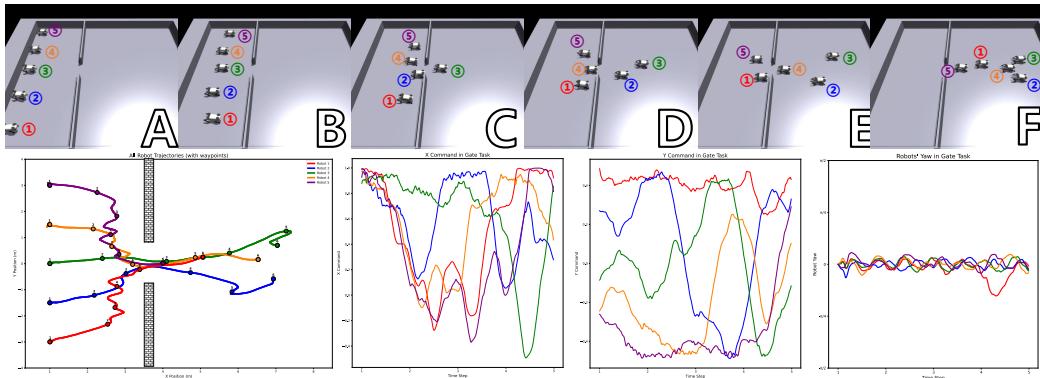

Figure 18: Visualization of the learned behaviors on `Gate` with 5 robots.

Across agents, the X Command traces show pronounced **wave-like** undulations: troughs denote the robot waiting at the gate, whereas crests indicate the robot currently traversing the passage. Around Frame B ($t\approx2$), an implicit queue emerges in the "X Command" panel: Robot 3 sustains a near-unity forward command while Robot 1 and Robot 2 reduce theirs moderately; Robot 4 and Robot 5 exhibit the deepest dips, preventing congestion at the bottleneck. In Frames (C-D) ($t=3\rightarrow4$), Robot 2 clears the gate first and issues a negative y-command to drift, vacating space for others. Robot 1 then accelerates forward to pass second, slightly shifting upward in y to prevent interference, followed by Robot 3. Finally, Robot 4 and Robot 5, which had consistently yielded earlier, complete the crossing. This progression underscores role specialization, with some agents acting as initiators and others as supporters, while the group achieves a well-aligned, collision-free sequence (*order: 3 → 2 → 1 → 4 → 5*).

C.3 ADDITIONAL BEHAVIOR VISUALIZATION

For further validation of the effectiveness of SeqWM, we provide additional visualizations of the learned behaviors on other tasks in Figure 19, Figure 20 and Figure 21.
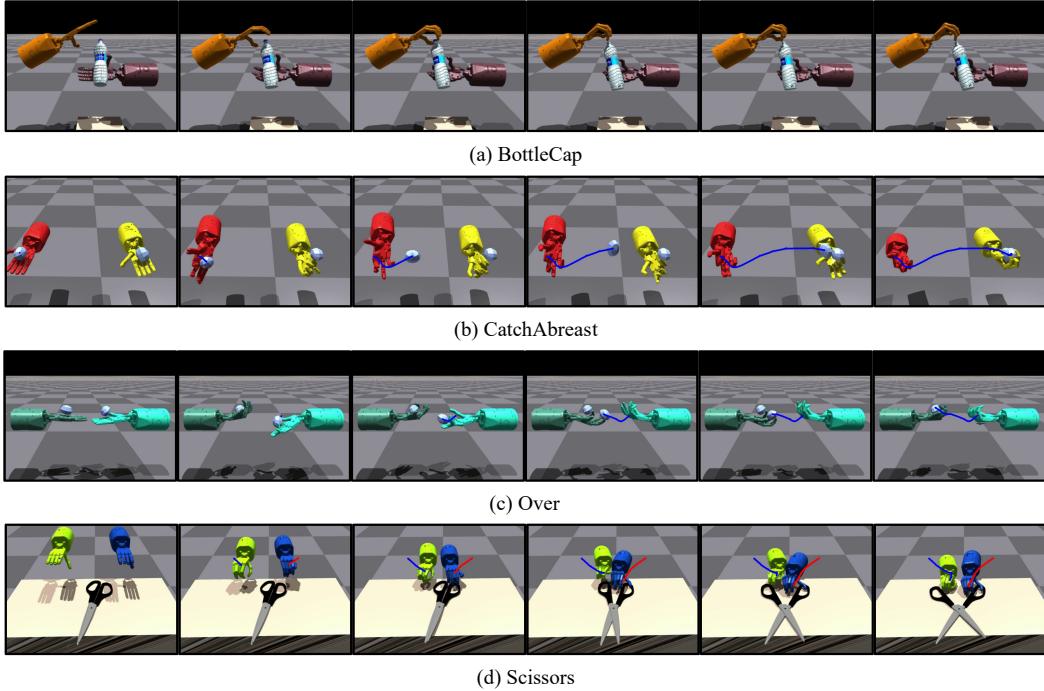

(a) BottleCap

(b) CatchAbreast

(c) Over

(d) Scissors

Figure 19: Visualization of the learned behaviors on the Bi-DexHands tasks.

For BottleCap, SeqWM learns effective division of labor and cooperation, with one hand stably grasping the bottle body while the other rotates and successfully unscrews the cap without tilting or dropping the bottle. For CatchAbreast, both hands successfully catch the object in parallel positions. The hand responsible for catching starts moving to the right before the object falls, adjusting its position to make the catching process more stable. Overall, these visualizations indicate that SeqWM can learn stable policies in high-dimensional state and action spaces, achieving advanced cooperative behaviors that surpass baseline methods through sequential communication, enabling early prediction, role division, temporal alignment, and intent sharing.
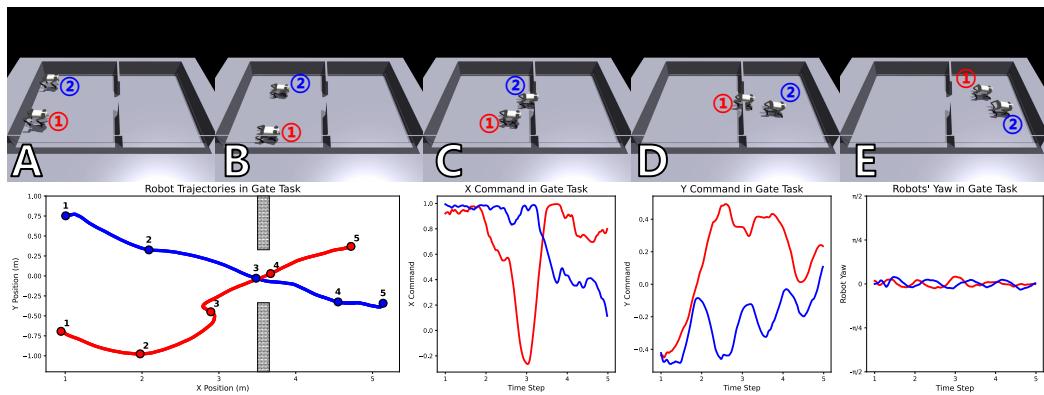

Figure 20: Visualization of the learned behaviors on Gate.

In Gate, from $t = 1$ (Frame A) to $t = 2$ (Frame B), both robots accelerate toward the narrow passage with high x-velocity commands. Around $t \approx 3$ (Frame C), Robot 1 reduces its x-command while

22

Robot 2 maintains forward motion, implicitly yielding priority to avoid collision. From $t{=}3$ (Frame C) to $t{=}4$ (Frame D), Robot 2 clears the gate and issues a negative y-command to move downward and create space (panel "Y Command"), while yaw remains near zero for both robots, indicating stable headings (panel "Robots' Yaw"). Finally, from $t{=}4 \rightarrow 5$ (Frame D $\rightarrow$ E), Robot 1 proceeds through the gate as Robot 2 decelerates, completing the task with smooth, collision-free coordination.
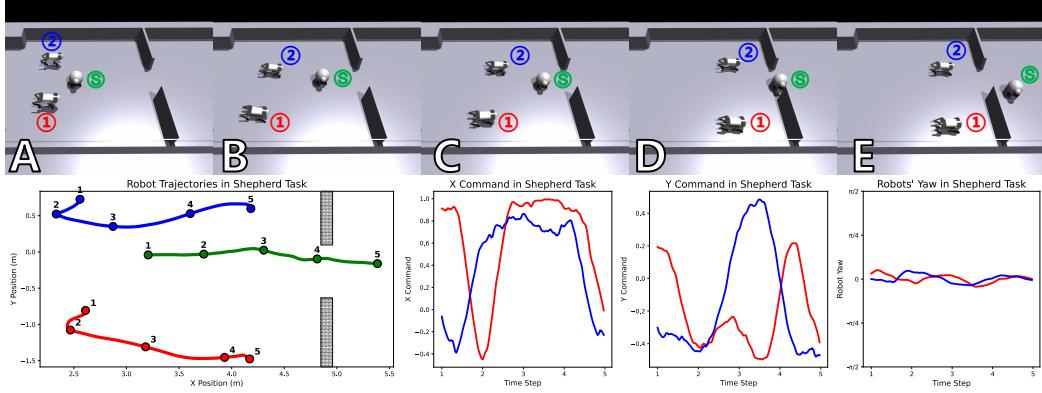

Figure 21: Visualization of the learned behaviors on `Shepherd`.

For the `Shepherd`, from $t = 3$ (Frame C) to $t = 4$ (Frame D), the sheep approaches the upper boundary. At this point, Robot 1 moves downward while Robot 2 advances to prevent the sheep from escaping, effectively guiding it back to the target area. This sequence highlights SeqWM's predictive planning and adaptive role assignment: the leading robot influences the trajectory of the target robot, while the following robot adjusts accordingly to ensure successful shepherding.

C.4 ROLE OF ACTOR AND PLANNER

As shown in Eq. (2), the world model consists of both an actor and a critic, but their roles are different. The actor is only used to generate candidate actions for the planner and does not directly participate in decision-making as an explicit policy. In contrast, the critic is crucial because it estimates the value of candidate actions in Eq. (7) and guides the update of the action distribution in Eq. (8).

Table 4 presents the results of ablation studies that clearly demonstrate the complementary roles of the planner and actor. The *planner-only* setting, where the planner searches over randomly sampled actions without leveraging the estimates from the actor, achieves moderate performance across tasks but suffers from inefficiency and suboptimal exploration, as reflected in the consistent performance drop compared to the full *planner-actor* configuration. Conversely, the *actor-only* variant, which directly executes actions proposed by the actor without deliberation from the planner, performs even worse, indicating that the actor lacks the capability to independently handle task-level reasoning or long-term coordination. The *planner-actor* combination yields the best results in all tasks, showing that the actor contributes by narrowing the action search space toward promising regions, while the planner ensures robust long-term decision-making and correction of suboptimal actor proposals. This synergy allows the system to balance efficiency and accuracy, highlighting that the actor serves primarily as a prior for action generation, whereas the planner is responsible for structured search and task-level optimization.

C.5 EARLY-STOPPING PLANNER

Inference Time Cost. We report the per-step execution time of SeqWM on `BottleCap` using a single RTX A6000 GPU on the left side of Figure 22. The execution time increases almost linearly with the rollout horizon H and the number of planner iterations K, which is consistent with the design of SeqWM. With the default settings, SeqWM achieves a per-step execution time of 12.8 ms, making it suitable for most real-time robotic tasks.

Early-Stopping Heuristic. To further enhance the efficiency of SeqWM, we introduce an early-stopping heuristic in Section 4.2 that terminates iterations when the change in the action distribution

Table 4: Ablation studies of actor.

Task	Planner-Actor	Planner-Only	Actor-Only
BottleCap	**480.4 ± 2.8**	420.3 ± 15.8	372.5 ± 9.8
CatchAbreast	**47.9 ± 0.8**	43.0 ± 1.3	40.6 ± 1.0
Over2Underarm	**30.5 ± 0.3**	28.4 ± 0.9	27.5 ± 0.7
Over	**50.1 ± 0.4**	46.1 ± 1.5	30.5 ± 1.1
Pen	**221.5 ± 2.2**	190.1 ± 6.6	163.8 ± 5.0
Scissors	**686.3 ± 3.1**	634.6 ± 22.1	592.1 ± 17.1

is not significant. The KL divergence is used as a measure of distribution change, and the execution time and performance under different thresholds on `BottleCap` are shown on the right side of Figure 22. When the threshold is set to 0.5, SeqWM reduces the execution time by approximately 27.3% while incurring only about 5.9% performance loss.
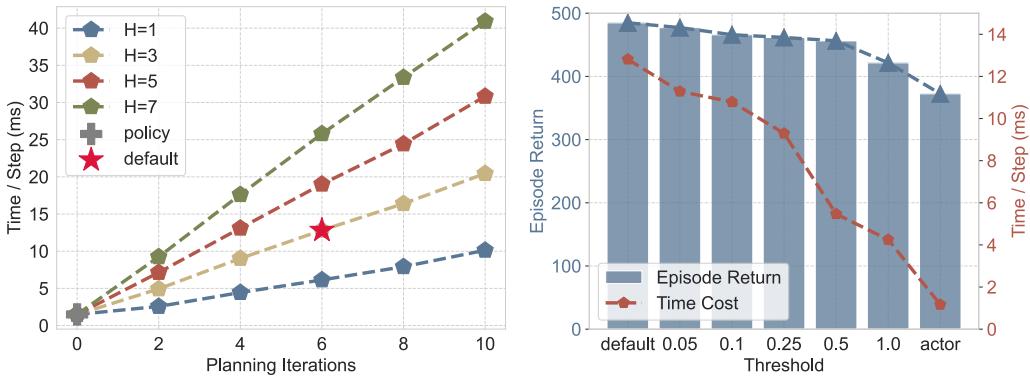

Figure 22: The per-step execution time of SeqWM. (*Left*): Time cost under different rollout horizons H and planner iterations K. (*Right*): Time cost and performance with and without early-stopping heuristic.

C.6 SIM-TO-REAL DEPLOYMENT

We implement all three Multi-Quad tasks in an $8\text{m} \times 5\text{m}$ indoor space. Each task involves two Unitree Go2-W quadruped robots. The room is equipped with eight Mars cameras, and real-time localization of robots and objects is provided by the NOKOV 3D motion capture system.

- **PushBox**: We use a cardboard box of $1.2\text{m} \times 1.2\text{m} \times 0.5\text{m}$ and approximately $6\,\text{kg}$ in mass. The box is sufficiently large that a single robot cannot independently control its movement direction, making cooperation essential. The static and kinetic friction coefficients between the box and the ground are both approximately 0.5.

- **Gate**: A 1m-wide doorway is set up. As shown in Figure 7 (b)-A, the two robots cannot pass through side-by-side, requiring coordinated navigation.

- **Shepherd**: A DJI EP robot acts as the guided agent (sheep). It is equipped with an omnidirectional chassis to simulate sheep behavior: it moves away from the nearest herding robot and its speed is inversely proportional to the distance to that robot.

We employ the following strategies to enhance the generalization capability of SeqWM and facilitate sim-to-real transfer:

- **Observation transformation**: Positions of other robots are transformed from the global frame into the ego-centric frame of the current robot, reducing observation complexity and improving policy generalization.
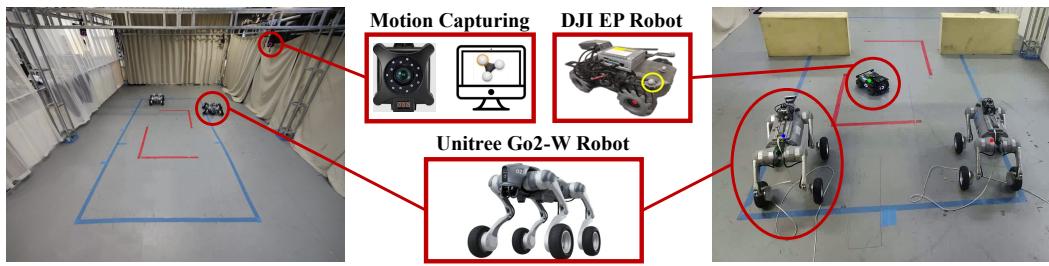
Figure 23: Real-world setups.

- **Domain randomization**: Taking `PushBox` as an example, we randomize the initial positions/orientations of both robots and the box, the position and distance of the target, and the friction coefficient between the box and floor to improve robustness to environmental variations.

- **Sensor and actuation perturbations**: Random noise is added to sensor readings, and small delays with noise are introduced into control commands to emulate real-world sensing errors and actuation inaccuracies.