AdaWM: Few-Shot Adaptation of World Models to Unseen Dynamical Regimes

Zian Guan[*]
Zhejiang University
Hangzhou, China
guanzian@zju.edu.cn

Guozheng Li[*†]
Southeast University
Nanjing, China
gzli@seu.edu.cn

Zilun Zhang
Zhejiang University
Hangzhou, China
zilun.zhang@zju.edu.cn

Zecong Tang[†]
Zhejiang University
Hangzhou, China
zecongt0410@gmail.com

Abstract

World models predict future states from observations and actions, enabling planning and decision-making in complex environments. Most existing approaches assume fixed environment dynamics and struggle to generalize to unseen regimes without retraining. However, in many real-world multi-agent systems, predictive dynamics vary across scenarios due to diverse styles and evolving agent behaviors. Meanwhile, only a small number of demonstration transitions are available in new environments, making adaptation to such unseen dynamical regimes challenging. In this work, we introduce *AdaWM*, a world model that achieves zero-gradient few-shot adaptation to unseen dynamical regimes through Feature-wise Linear Modulation (FiLM). *AdaWM* is built on a JEPA-style latent prediction framework with a permutation-invariant context encoder whose output modulates the model through FiLM, enabling fast adaptation without modifying model parameters. Across heterogeneous multi-agent environments, *AdaWM* consistently reduces prediction error under distribution shift, outperforming fine-tuning and MAML in both adaptation gain and efficiency. Specifically, it achieves up to 15% reduction in prediction error with only $K{=}5$ demonstrations. Moreover, we show that few-shot adaptation complements model scaling: explicit context conditioning remains effective at larger model capacities, including the LeWM architecture, without requiring test-time parameter updates. These results show that *AdaWM* provides an effective approach for adapting world models to novel dynamical regimes.

1 Introduction

World models are learned simulators that predict future states given current observations and actions [53, 59]. By learning compact latent dynamics, neural world models enable agents to "dream" within a learned transition model before acting, reducing costly real-world trial-and-error [21–23, 25]. Recent advances have extended this idea to pixel-based game simulators [31, 72] and representation-space predictors [4, 9, 34], with the Latent Exploration World Model (LeWM) showing that JEPA-based architectures can be both effective and parameter-efficient by predicting task-relevant latent representations rather than reconstructing pixel-level details [38]. However, these models typically

[*]Equal contribution.
[†]Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

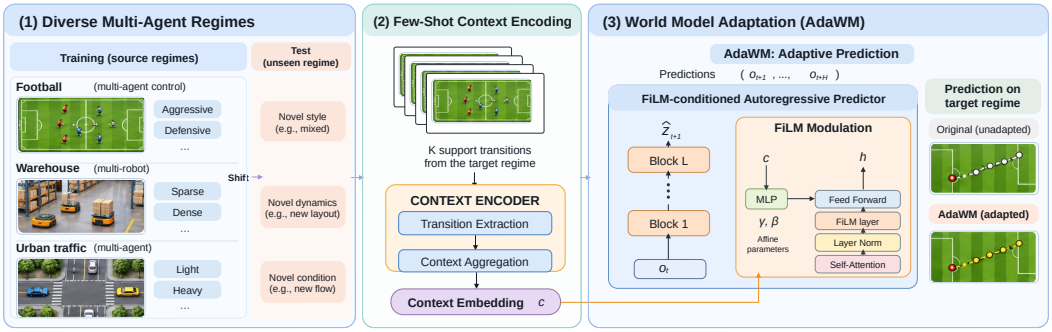


Figure 1: Multi-agent systems exhibit diverse dynamical regimes. Rather than learning per-pattern models or fine-tuning for each new behavior pattern, *AdaWM* treats few-shot adaptation as conditional inference. A few target transitions define a style context that modulates the latent BEV predictor.

learn a single transition law under a fixed-dynamics assumption, where test-time dynamics are expected to follow the same regime as training [3, 24, 41].

This representation-space perspective is particularly useful for adaptation: focusing on predictive latent features rather than low-level observation details helps separate stable transition structure from scenario-specific variations. In multi-agent environments, however, the fixed-dynamics assumption still breaks down because the same observation space can exhibit different transition dynamics as interaction patterns change across scenarios [2, 43]. We refer to these scenario-specific interaction patterns as *dynamical regimes*. Taking sports as an example, a model trained on short-passing ("tiki-taka") dynamics produces inaccurate predictions under counter-attacking regimes with rapid transitions [32, 61, 66, 69]. More generally, in multi-agent systems such as robotics and traffic, dynamics are shaped by collective agent behaviors, which differ in coordination patterns and interaction intensity [29, 52]. Adapting to a new regime is particularly challenging in the few-shot setting, where limited demonstrations are available and gradient-based methods tend to overfit [44, 75].

Existing solutions typically require retraining or maintaining multiple models, which is computationally expensive and does not scale. The core question we address is: *Can a world model adapt to unseen multi-agent dynamical regimes from only a handful of demonstrations, without any gradient updates to the model parameters?*

We answer affirmatively by introducing *AdaWM*, a few-shot JEPA world model that conditions its predictions on a small context set through FiLM [46]. The key insight is that FiLM parameters (γ, β) act as *implicit dynamics parameters*: the context vector $\boldsymbol{c}$ encodes "what kind of multi-agent behavior to expect," adjusting the predictor's computations without modifying its weights. Critically, while the context encoder is learned offline, no gradient computation occurs at test time: adaptation is a single forward pass through the context encoder, making it orders of magnitude faster than gradient-based alternatives like fine-tuning or MAML [18]. Our contributions are threefold:

- **Few-shot adaptation for multi-agent world models.** We study the problem of adapting world models to unseen multi-agent dynamical regimes from a small number of demonstration transitions, where environment dynamics vary across scenarios and adaptation data is scarce.
- **An efficient zero-gradient adaptation mechanism.** We instantiate *AdaWM* as a JEPA-based world model with FiLM conditioning and a permutation-invariant context encoder, enabling domain-agnostic adaptation to multi-agent dynamics in one forward pass without test-time optimization.
- **Empirical insights on adaptation and scale.** Across diverse multi-agent settings, we show that few-shot conditioning achieves consistent gains with limited demonstrations, complements model scaling, and benefits from larger distribution shifts between training and target regimes.

2 Related Work

World Models and JEPA Architectures. World models enable sample-efficient planning by learning an internal environment simulator, from early Dyna-style planning [53, 59] to modern latent formulations [21, 25, 31, 54]. Recent pixel-based simulators, such as IRIS [41], STORM [72], and

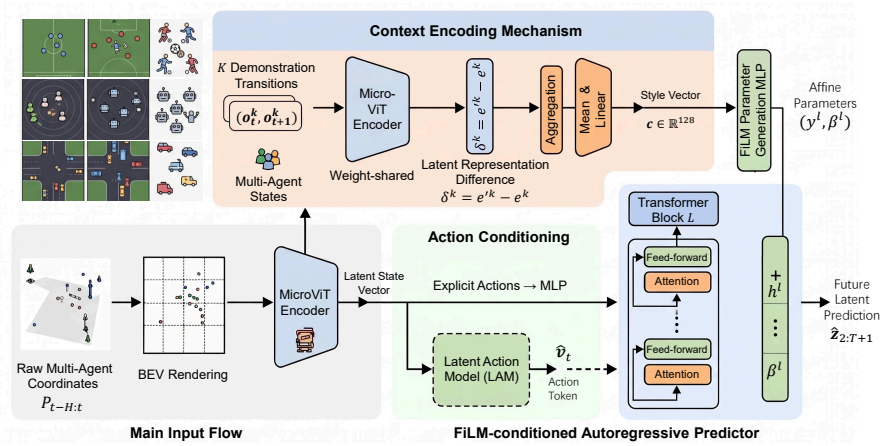


Figure 2: **Architecture overview.** *AdaWM* utilizes a MicroViT to encode BEV-rendered observations and a context encoder to generate a style vector that modulates an autoregressive predictor via FiLM. The entire system contains 4.55M parameters and trains in under an hour on a consumer GPU.

DIAMOND [3], further scale these dynamics using transformers and diffusion [13]. Building on this, the Joint Embedding Predictive Architecture (JEPA) [34] avoids pixel-level reconstruction via latent prediction [4, 9], utilizing regularizers like Barlow Twins [71], VICReg [8], or SIGReg [7] to prevent representation collapse. This latent prediction paradigm has recently been adopted in world models learned directly in representation space. For instance, LeWM [38] demonstrates stable training from pixels, while related works explore latent spaces for multi-agent RL [74] or single-agent adaptation [37]. However, most existing world models still assume fixed environment dynamics and lack mechanisms for test-time adaptation. *AdaWM* bridges this gap by extending JEPA with in-context conditioning for adaptation to unseen behavioral regimes [33, 39, 47].

Few-Shot Adaptation and In-Context Learning. Gradient-based meta-learning, such as MAML [18] and Meta-RL variants, enables few-shot adaptation but incurs high test-time computational costs [17, 44, 48, 75]. Therefore, in-context learning (ICL) [12] has emerged as a powerful zero-gradient alternative, with theoretical foundations showing that transformer forward passes can implicitly perform learning algorithms [1, 67] or gradient descent [65, 73]. For lightweight conditioning, Feature-wise Linear Modulation (FiLM) [46] remains an effective affine modulation mechanism [70]. Related work on context-aware policies and contextual dynamics models has also explored learning environment-specific representations for generalization [11, 35, 47, 50], with broader connections to in-context reinforcement learning [42]. While prior JEPA models have successfully explored few-shot classification [4] and video understanding [9], to our knowledge, few prior works study zero-gradient few-shot adaptation in JEPA *world models* for multi-agent dynamical prediction.

Multi-Agent Dynamics Modeling. Multi-agent trajectory prediction is extensively studied in autonomous driving [29, 43, 52, 56] and pedestrian forecasting [2, 19, 20, 28], with recent works like SingularTrajectory [6] unifying adaptation tasks via diffusion. In sports analytics, Google Research Football (GRF) [32] provides a foundational physics-based simulator. Building on such tracking data, task-specific models like RoleFor and TacticAI achieve state-of-the-art results on trajectory forecasting [55, 60, 66, 69], while AdaSports-Traj [68] introduces domain-aware adaptation. However, these methods typically operate in *coordinate space* with highly specialized, task-specific architectures [14, 61]. In contrast, *AdaWM* takes a different approach: we learn a domain-agnostic world model in latent space that adapts to unseen behavioral patterns through few-shot conditioning, seamlessly generalizing across various domains with a single unified architecture.

3 Method

We propose *AdaWM*, a world model for few-shot adaptation to unseen multi-agent dynamics. The method comprises four key components (Figure 2): (1) a spatial representation that converts agent coordinates into latent states via **BEV rendering** and a lightweight **MicroViT encoder**; (2) an

action conditioning module that provides action embeddings or inferred latent actions; (3) a **context encoding mechanism** that aggregates demonstration transitions into a compact representation of target dynamics; and (4) a **FiLM-conditioned autoregressive predictor** that adapts its dynamics prediction based on this context, with an optional **BEV decoder**. We detail each component below.

3.1 Problem Formulation

We consider a multi-agent environment with N agents whose positions at time t are $\boldsymbol{p}_t = \{(x_i^t, y_i^t)\}_{i=1}^N$. A world model predicts future observations given past observations and actions:

$$\hat{\boldsymbol{o}}_{t+1} = f(\boldsymbol{o}_{t-H:t}, \boldsymbol{a}_{t-H:t}). \tag{1}$$

In the *few-shot adaptation* setting, in addition to historical observations $\boldsymbol{o}_{t-H:t}$ and actions $\boldsymbol{a}_{t-H:t}$, we receive a small context set of K demonstration transitions $\mathcal{C} = \{(\boldsymbol{o}_t^k, \boldsymbol{o}_{t+1}^k)\}_{k=1}^K$ from a target environment with potentially different agent dynamics. Our goal is to condition predictions on $\mathcal{C}$ *without updating model parameters*:

$$\hat{\boldsymbol{o}}_{t+1} = f(\boldsymbol{o}_{t-H:t}, \boldsymbol{a}_{t-H:t} \mid \mathcal{C}). \tag{2}$$

3.2 Spatial Representation

Raw multi-agent tracking data consists of positions for multiple agents (and optional objects) at each timestep. Rather than operating on variable-size coordinate vectors, we decouple the architecture from entity count by rendering each frame as a fixed-size bird's-eye-view (BEV) image. We encode each BEV frame $\boldsymbol{o}_t$ into a latent state vector:

$$\boldsymbol{z}_t = f_\theta(\boldsymbol{o}_t) \in \mathbb{R}^d, \quad d = 128, \tag{3}$$

using a lightweight Vision Transformer [16]. This approach preserves spatial locality, enabling attention-based architectures to naturally capture local interactions. Furthermore, it ensures that any multi-agent system ranging from sports to robotics and traffic settings can be processed by the identical architecture, while providing an interpretable visualization space (see Figure 3).

For visualization and qualitative evaluation, we design an optional BEV decoder that reconstructs images from latent vectors: $\hat{\boldsymbol{o}}_t = \text{Dec}(\boldsymbol{z}_t)$. This is not used during prediction but only for rendering predicted latent states into interpretable images. Full details are provided in Appendix I.

3.3 Action Conditioning

For simulated environments with explicit action labels, we encode actions using a lightweight MLP. Discrete actions are embedded via a two-layer MLP: $\boldsymbol{a}_t = \text{MLP}(\text{onehot}(a_t)) \in \mathbb{R}^{128}$.

For real tracking data lacking explicit action labels (e.g., sports broadcasts), we replace this with a latent action model. Specifically, we use a VQ-VAE-based Latent Action Model [13] that infers a discrete action token a_t from consecutive latent states:

$$\hat{\boldsymbol{v}}_t,\ a_t = \text{VQ}\left(\text{MLP}_{\text{infer}}([\boldsymbol{z}_t; \boldsymbol{z}_{t+1}]);\ \mathcal{C}_{32}\right), \tag{4}$$

where $\mathcal{C}_{32}$ is a learnable 32-entry codebook. The resulting token $\hat{\boldsymbol{v}}_t$ replaces the explicit action embedding $\boldsymbol{a}_t$ in the predictor. Full architectural and training details are given in Appendix D.

3.4 Few-Shot Adaptation Mechanism

The autoregressive predictor forecasts future latent states conditioned on past states, actions, and the target style context vector:

$$\hat{\boldsymbol{z}}_{2:T+1} = g_\phi(\boldsymbol{z}_{1:T}, \boldsymbol{a}_{1:T}, \boldsymbol{c}). \tag{5}$$

Context Encoding. Given K demonstration transition pairs $\{(\boldsymbol{o}_t^k, \boldsymbol{o}_{t+1}^k)\}_{k=1}^K$ from the target environment, we extract the initial state, next state, and their transition difference. This captures both the static state distribution (where agents are) and the transition dynamics (how they move). We encode each transition into latent representations:

$$\boldsymbol{e}^k = f_\theta(\boldsymbol{o}_t^k), \quad \boldsymbol{e}'^k = f_\theta(\boldsymbol{o}_{t+1}^k), \quad \boldsymbol{\delta}^k = \boldsymbol{e}'^k - \boldsymbol{e}^k. \tag{6}$$

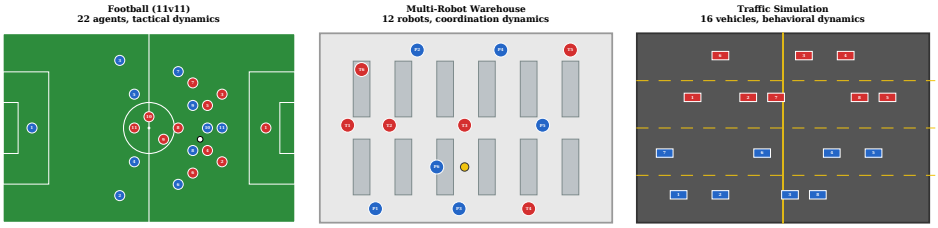


Figure 3: **Domain-agnostic BEV rendering.** Multi-agent dynamics vary dramatically across behavioral styles. The same rendering pipeline and model architecture are applied to three fundamentally different multi-agent domains. No architectural changes are needed, only the data source differs.

Specifically, we include the difference vector $\boldsymbol{\delta}^k$ explicitly to encode the *direction and magnitude of state change* between consecutive frames. Without it, the context encoder would need to implicitly compute dynamics from two static snapshots. Providing this direct dynamics signal is important when styles differ primarily in *motion* rather than *spatial layout*. We then aggregate these transition descriptors into a permutation-invariant context vector:

$$\boldsymbol{c} = \frac{1}{K} \sum_{k=1}^{K} \text{MLP}\big([\boldsymbol{e}^k; \boldsymbol{e}'^k; \boldsymbol{\delta}^k]\big) \in \mathbb{R}^{128}. \tag{7}$$

When $K = 0$ (no demonstrations available), we set $\boldsymbol{c} = \mathbf{0}$, making the predictor unconditional.

FiLM Conditioning. We condition the predictor using Feature-wise Linear Modulation (FiLM), where the context vector $\boldsymbol{c}$ generates affine parameters for each transformer block:

$$(\boldsymbol{\gamma}^l, \boldsymbol{\beta}^l) = \text{MLP}^l(\boldsymbol{c}) \in \mathbb{R}^{2D}, \tag{8}$$

which modulate hidden states:

$$\boldsymbol{h}^{l+1} = \boldsymbol{\gamma}^l \odot \boldsymbol{h}^l + \boldsymbol{\beta}^l. \tag{9}$$

Acting as an *implicit parameterization* of environment dynamics, this modulation uses γ values to amplify context-specific features. Unlike gradient-based meta-learning, this mechanism achieves zero-gradient adaptation within a single forward pass.

3.5 Training Objective

We train the model end-to-end using a combined objective consisting of a latent prediction loss, a regularization term, and an optional latent action loss when action labels are unavailable:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda \cdot \mathcal{L}_{\text{SIGReg}} + \beta \cdot \mathcal{L}_{\text{LAM}}. \tag{10}$$

Prediction loss. We minimize the mean squared error between predicted and target latent states in a shared projection space:

$$\mathcal{L}_{\text{pred}} = \frac{1}{T} \sum_{t=1}^{T} \left\| \text{proj}_{\text{pred}}(\hat{\boldsymbol{z}}_{t+1}) - \text{sg}\left[\text{proj}_{\text{tgt}}(\boldsymbol{z}_{t+1})\right] \right\|^2, \tag{11}$$

where $\text{sg}[\cdot]$ denotes stop-gradient on the target branch, following JEPA conventions.

SIGReg regularization. To further stabilize representation learning, we apply SIGReg [7], which enforces an isotropic Gaussian structure in the latent space via characteristic function matching.

Training protocol. During each iteration, we sample $K{=}5$ demonstrations from a random training style to compute $\boldsymbol{c}$, while predicting a separate target sequence from the same style. All details are provided in Appendix I.7.

4 Experiments

We evaluate *AdaWM* along the following four key questions: (1) Does *AdaWM*'s FiLM-based conditioning enable effective few-shot adaptation of multi-agent dynamics? (2) How does it compare to gradient-based and non-parametric baselines? (3) Does the learned model produce stable long-horizon rollouts? (4) Can the approach generalize across real-world data and diverse domains?

Table 1: **Main comparison**. Prediction MSE ± std over 100 test episodes, averaged across 5 random seeds. ↓ lower is better. **Bold**: best within each adaptation category. Shaded : best zero-gradient adaptation. At K=5, *AdaWM* achieves -15.3% (CA) and -9.9% (HP) mean adaptation gain.

		Adaptation	*counter_attack*		*high_press_chaos*	
Category	Method	Cost	MSE ↓	Std	MSE ↓	Std
No adaptation (fixed model):	No-Context (K=0)	none	0.555	0.046	0.657	0.052
	LeWM [38] (K=0, 17.9M)	none	0.331	0.092	0.421	0.158
Gradient-based adaptation:	Fine-Tune (K=5, 10 steps)	10 grad steps	0.544	0.043	0.644	0.054
	MAML (K=5, 5 inner steps)	5 grad steps	0.545	0.039	0.651	0.056
	LeWM + FT (K=5, 5 steps)	5 grad steps	0.464	0.162	0.577	0.160
Zero-gradient adaptation (Ours):	*AdaWM* (K=5, **Ours**)	**0 grad steps**	**0.470**	0.042	**0.592**	0.057
	AdaWM (K=10, **Ours**)	**0 grad steps**	**0.469**	0.036	**0.599**	0.066
Without action labels (LAM-inferred actions):	*AdaWM* + LAM (K=0)	0 grad steps	0.160	0.041	0.163	0.039
	AdaWM + LAM (K=5)	0 grad steps	0.178	0.031	0.188	0.042

4.1 Experimental Setup

Environments and Data. We evaluate *AdaWM* across 17 diverse multi-agent environments. Our primary testbed consists of two football simulators (*v1* and *v2*) designed to generate controlled distribution shifts across 8 tactical styles with varying team compactness and player dynamics. To assess sim-to-real and cross-domain generalization, we utilize real-world football tracking (Metrica [40], SkillCorner [57]) and trajectory benchmarks (ETH/UCY [36, 45], NGSIM [62]), alongside multi-agent warehouse and traffic simulators. We further test sport-agnostic transfer using basketball, badminton, and tennis environments. Full details are provided in Appendix N.

Model configuration and Baselines. All experiments use a MicroViT encoder (d=128, 4 layers) and a FiLM-conditioned autoregressive predictor (D=256, 3 layers), totaling 4.55M parameters. The system enables real-time 60 FPS prediction (16.5 ms/step) and trains within 1 hour (60 epochs) on a single NVIDIA RTX 2060 Super. We compare against: **No-Context** (K=0) measures unconditional prediction quality; a **GRU Predictor** replaces the FiLM transformer with a 3-layer GRU to test the architectural impact; **Fine-Tuning (FT)** performs 10 gradient steps on the predictor using K demos; **MAML** applies 5 inner-loop meta-gradient steps to test the strongest gradient-based alternative; and **Nearest Neighbor (NN)** provides a non-parametric retrieval baseline based on latent L2 distance.

Evaluation protocol. For few-shot experiments, we adopt the standard N-way-K-shot episodic evaluation protocol [58]. Each test episode samples K support transitions from one held-out style, then evaluates prediction MSE on a separate query set from the same style, ensuring all methods observe *exactly* K examples before predicting. We compare FiLM against fine-tuning (gradient updates) per episode. The K=0 baseline uses zero-vector context (FiLM active but uninformative).

4.2 Few-Shot Dynamics Adaptation

Baseline Comparison. We first evaluate few-shot adaptation on the v2 football simulator. Table 1 presents results where models are trained on 6 styles and evaluated on 2 held-out styles. We further verify that this result is not an artifact of the specific split via leave-one-style-out cross-validation across all 8 styles (Appendix J): FiLM improves 6/8 held-out styles with mean gain -3.6%.

Gradient-based adaptation fails in the small-K regime. Fine-tuning and MAML under the same base architecture consistently underperform *AdaWM* despite using test-time gradient updates. With only K=5 demonstrations, gradient-based methods capture limited style statistics and overfit rapidly; Appendix G shows that additional updates eventually degrade performance. On *counter_attack*, K=5 reduces MSE by 15.3% (0.555→0.470), demonstrating that a lightweight conditioning mechanism can effectively extract useful information from only a handful of demonstrations.

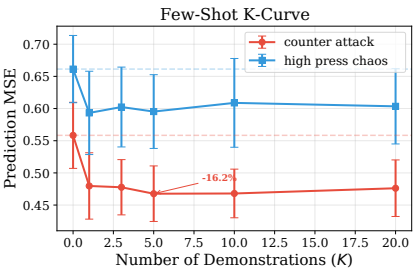


Figure 4: Few-shot K-curve (sim).

Adaptation complements model scaling. At 4.55M parameters, *AdaWM* is ∼3.9× smaller than LeWM [38] (17.9M). LeWM achieves lower absolute

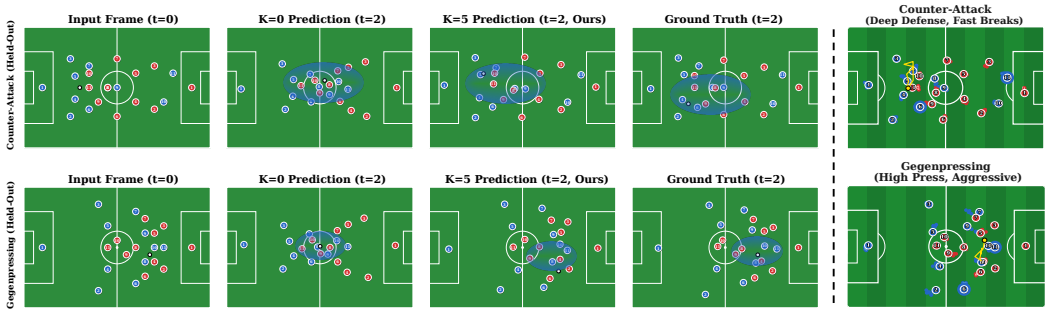


Figure 5: **Few-shot adaptation visualization.** For two held-out styles (rows), we show: input frame, $K{=}0$ prediction, $K{=}5$ prediction, and ground truth. The $K{=}5$ predictions visibly match the target.

zero-shot MSE through model capacity alone, but fine-tuning on $K{=}5$ demonstrations degrades performance by $+40\%/+37\%$ in our experiments. Further matched-capacity experiments show that context conditioning remains effective at larger model scales, including LeWM, indicating that adaptation and model scaling are complementary. In an ego-only setting, *AdaWM* further reduces prediction error by $7.8\%/7.1\%$, demonstrating effective adaptation when other agents' controls are unobserved. Detailed analysis is reported in Appendix I.12. Furthermore, as shown in Figure 4, most adaptation gains are achieved with very few examples: $K{=}1$ nearly already provides a substantial improvement, confirming that *AdaWM* operates in a true few-shot regime.

LAM provides an action-free modeling pathway. Using LAM-inferred actions substantially lowers prediction error (0.160/0.163 vs. 0.555/0.657) by learning a compressed latent action space. However, FiLM adaptation depends on style-discriminative action representations. We analyze in Appendix D.

Episodic Few-Shot Evaluation. To more rigorously evaluate few-shot adaptation, we adopt the standard N-way-K-shot episodic protocol [58]. Appendix C shows that FiLM achieves lower MSE than fine-tuning at every tested step budget, while fine-tuning overfits as steps increase. This confirms that gradient-based adaptation in the small-K regime is unstable, whereas FiLM provides a more robust adaptation mechanism. Simultaneously, for **total inference speed**, FiLM conditioning requires only 22 ms per episode, compared with 220 ms for FT-5, 738 ms for FT-20, and 3,671 ms for FT-100, making it 10–163$\times$ faster while also achieving better prediction quality.

Few-Shot Visual Comparison. Figure 5 provides qualitative evidence that FiLM conditioning adapts predicted dynamics. Given the same input frame, the $K{=}0$ predictor produces a generic trajectory resembling the training average, while the $K{=}5$ predictor aligns more closely with the ground truth of the held-out style. This is particularly evident in the spatial layout of players: the $K{=}5$ prediction correctly captures the stretched formation with strikers high and defenders deep of counter-attack play. We visualize the *velocity field* in Appendix I.11.

Table 2: **Real-world Football Evaluation.** *AdaWM* adapts while FT-5 fails to improve.

Dataset / Match	$K{=}0$	FT-5	*AdaWM*	Δ
Metrica Sports (Optical tracking, 25Hz)				
Metrica Game 1	0.469	0.468	**0.363**	-22.7%
Metrica Game 2	0.445	0.444	**0.307**	-31.0%
Metrica Average	**0.457**	0.456	**0.335**	$\mathbf{-26.7\%}$
SkillCorner (Broadcast, 10 A-League matches)				
Auckland vs Newcastle	0.349	0.349	**0.267**	-23.4%
CC Mariners vs Melb City	0.488	0.488	**0.377**	-22.7%
Sydney vs Adelaide	0.334	0.334	**0.252**	-24.8%
SkillCorner Avg (10)	**0.381**	0.379	**0.289**	$\mathbf{-24.1\%}$

4.3 Generalization to Real and Cross-Domain Data

We further evaluate *AdaWM* across real-world tracking data, and diverse multi-agent domains with varying interaction complexity. As summarized in Table 3, *AdaWM* consistently improves prediction accuracy in 17/19 settings under unified evaluation protocol, with an average reduction of 8.5% over the no-context baseline, while gradient-based adaptation provides only marginal gains.

Sim-to-Real Generalization We train *AdaWM exclusively* on synthetic data and evaluate it directly on real matches from Metrica and SkillCorner without fine-tuning, therefore the model must use support frame pairs to generalize from simulated dynamics to real tracking data.

Table 3: **Comprehensive cross-domain evaluation.** *AdaWM* evaluated across 7 domains, 4–100 agents, with $K{=}5$ demonstrations. Shaded: $\geq 5\%$ Ours improvement. *AdaWM* improves **17/19** (89%), mean $\Delta{=}-8.5\%$. Percentage changes are computed from unrounded MSE values.

				Prediction MSE ↓				Adapt. Δ vs. No-Ctx	
Domain	**Held-out Style**	**#Agent**	**Data**	No-Ctx	FT-5	MAML	**Ours**	FT-5	**Ours**
Synthetic multi-agent environments:									
Football (v2)	counter_attack	22	Sim	0.555	0.542	0.545	**0.470**	−2.3%	**−15.3%**
Football (v2)	high_press_chaos	22	Sim	0.657	0.645	0.651	**0.592**	−1.8%	**−9.9%**
Warehouse	random_walk	12	Sim	0.134	0.133	0.135	**0.132**	−0.7%	−1.4%
Warehouse	greedy	12	Sim	0.134	0.134	0.135	**0.132**	−0.3%	−1.5%
Traffic	aggressive	16	Sim	0.133	0.132	0.133	**0.132**	−0.5%	−1.2%
Traffic	rush_hour	16	Sim	0.133	0.133	0.134	**0.132**	−0.1%	−1.2%
Synthetic multi-sport:									
Basketball	zone_defense	10	Sim	0.065	0.064	0.065	**0.063**	−0.8%	−2.8%
Basketball	full_court_press	10	Sim	0.064	0.064	0.065	**0.062**	−0.4%	−2.8%
Badminton	mixed	4	Sim	0.044	0.044	0.045	**0.043**	−0.2%	−1.8%
Badminton	aggressive_serve	4	Sim	0.044	0.043	0.044	**0.043**	−0.6%	−1.8%
Tennis	counterpuncher	4	Sim	0.020	0.020	0.021	**0.020**	+0.2%	−0.1%
Tennis	all_court	4	Sim	0.021	0.021	0.021	**0.021**	+0.1%	−0.1%
Real-world† tracking data (zero-shot sim-to-real transfer):									
Metrica†	Game 1 (optical 25Hz)	22	Real	0.469	0.468	0.471	**0.363**	−0.2%	**−22.7%**
Metrica†	Game 2 (optical 25Hz)	22	Real	0.445	0.444	0.448	**0.307**	−0.1%	**−31.0%**
SkillCorner†	10-match avg. (bcast.)	22	Real	0.381	0.379	0.384	**0.289**	−0.5%	**−24.1%**
ETH/UCY†	zara01 (pedestrian)	5–40	Real	0.092	0.091	0.093	**0.078**	−0.8%	**−15.3%**
ETH/UCY†	zara02 (pedestrian)	5–40	Real	0.091	0.090	0.092	**0.078**	−0.7%	**−14.5%**
NGSIM†	dense-slow (congestion)	60–100	Real	0.014	0.014	0.015	**0.013**	−0.5%	**−6.0%**
NGSIM†	dense-fast (flowing)	60–100	Real	0.198	0.198	0.199	0.198	+0.1%	−0.3%
Adaptation latency (per episode)				—	220 ms	450 ms	**22 ms**	**Ours:** 10–20× **faster**	

As shown in Table 2, FiLM conditioning achieves substantial improvements of 17%–31% MSE reduction across all 12 real matches, while gradient-based adaptation (FT-5 with $\approx 0\%$) shows **no improvement**. FiLM conditioning bridges the sim-to-real gap, and we show details in Appendix I.9.

Cross-Domain Generalization We further evaluate *AdaWM* on diverse systems beyond football, including warehouse robotics, traffic simulation, pedestrian trajectories, and real traffic data.

Adaptation gains vary significantly across domains: complex interaction multi-agent settings such as pedestrian crowds achieve large improvements ($\sim$15%), while simpler systems such as traffic flow and warehouse coordination show smaller gains ($\sim$1–2%). On warehouse robotics (12 robots), few-shot conditioning yields $-1.4\%/-1.5\%$ MSE reduction. On traffic intersection (16 vehicles), $K{=}5$ achieves -1.2% improvement. Consistent with our gap-proportional principle, adaptation is most effective in complex "dense-slow" congestion (-6.0% on NGSIM) vs. uniform free-flow (-0.3%), confirming that *AdaWM* excels where inter-agent interactions are dense and style-dependent.

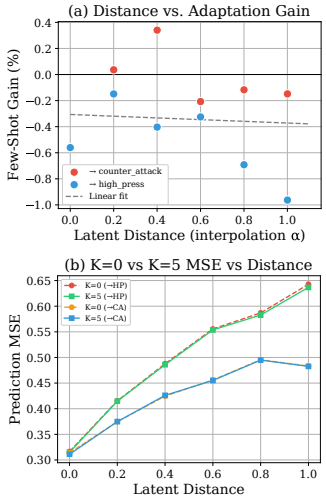


Figure 6: Distribution gap explains adaptation gain.

This pattern suggests that adaptation benefit is governed by interaction complexity rather than domain identity. Crucially, FiLM conditioning never degrades performance, indicating that it behaves conservatively when style signals are weak. Details are provided in Appendix I.9.

Multi-Sport Generalization Since the architecture is sport-agnostic, we evaluate *AdaWM* on basketball, badminton, and tennis. As summarized in Table 3, FiLM conditioning consistently improves prediction quality, with gains proportional to agent interaction complexity. Basketball shows a meaningful improvement (-2.8%), badminton shows moderate gains (-1.8%), and tennis is neutral (-0.1%) under the episodic protocol.

The magnitude of improvement correlates with both inter-style behavioral divergence and *agent density*: basketball (10 agents) exhibits the largest formation differences and the most interactions, while tennis (4 agents) has minimal inter-agent coordination to adapt. Football's 22 agents create rich interaction patterns that vary substantially across tactical styles; traffic and warehouse have fewer

agents and simpler interaction structures. This agent-density effect is consistent across all evaluated domains (Table 3). Crucially, FiLM conditioning never *hurts* prediction quality, confirming that the mechanism is conservative.

4.4 Why Does FiLM Work?

We analyze *why FiLM-based adaptation works* from three perspectives:

Why does FiLM outperform fine-tuning? The N-way-K-shot episodic evaluation (Section 4.2) shows that gradient-based adaptation overfits rapidly in the low-K regime, while FiLM remains stable across rollout horizons (e.g., FT-100 yields $+28.3\%$/$+25.3\%$ MSE degradation). FiLM conditioning compresses demonstrations into a single 128-d context vector $\boldsymbol{c}$ that modulates the predictor globally, without modifying model parameters. We provide a visual explanation in Appendix H. This behavior is consistent with in-context learning interpretations [65], where the context encoder effectively performs a single meta-update in a regularized feature space. In practice, FiLM achieves both improved generalization and a 10–163$\times$ speed advantage over gradient-based methods.

Distribution Gap Explains Adaptation Gain. A key insight is that *few-shot adaptation gain is proportional to the distribution gap* between training and target dynamics. Figure 6 shows a clear correlation between the latent cosine distance of a held-out style from its nearest training style and the improvement from K. Styles close to the training distribution (*tiki_taka*, distance 0.08) show negligible gain, while distant styles (*counter_attack*, distance 0.60) show measurable improvement. This also explains the real-data results: professional matches from the same data source exhibit very small latent gaps ($\sim$0.02), leading to strong zero-shot performance and limited additional gain. Together, FiLM adaptation is most beneficial under significant distributional shift.

Counterfactual Evaluation. The preceding analyses show that FiLM-based adaptation works by providing a regularized conditioning mechanism that selectively adjusts dynamics when distributional shift is present, while remaining stable otherwise.

To further verify that FiLM conditioning modulates dynamics rather than merely rescaling outputs, we design a *counterfactual experiment*: we condition the same input sequence on demos from 4 different styles and measure prediction divergence.

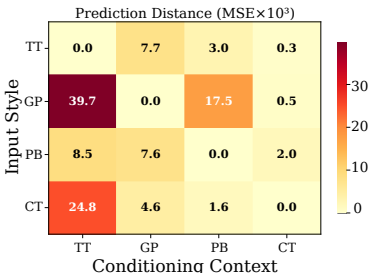


Figure 7: Distance heatmap. TT: tiki-taka; GP: gegenpressing; PB: park-the-bus; CT: counter-attack.

Figure 7 shows the pairwise distances between predictions conditioned on different styles. The mean off-diagonal distance is **9.82**, indicating FiLM conditioning produces substantially different predicted trajectories depending on the context. The largest distance (39.67) is between gegenpressing and tiki_taka contexts, consistent with opposing tactical philosophies (high-intensity pressing vs. slow possession buildup), confirming the context vector encodes meaningful information.

World Model Validation. A key question is whether *AdaWM* functions as a true world model rather than a one-step predictor. We validate this through three complementary aspects: autoregressive long-horizon rollout stability, trajectory-level consistency and planning, all of which indicate stable multi-step dynamics and coherent predictions. Full details are provided in Appendix E.

4.5 Ablation Studies

FiLM conditioning. Disabling FiLM eliminates all adaptation benefit in our architecture (Table 4). A matched six-way fusion comparison further shows that FiLM achieves stronger relative adaptation gains with fixed-size context, while cross-attention obtains lower absolute prediction error (Appendix I.13).

Table 4: **Ablation studies.** Eval on simulator.

Variant	K=0	K=5	Gain
Full (w/ FiLM + SIGReg)	0.555	**0.470**	-15.3%
w/o FiLM ($\boldsymbol{c}$=**0**)	0.555	0.555	0.0%
w/o SIGReg (λ=0)	0.383	0.383	0.0%

SIGReg. Without SIGReg ($\lambda = 0$), latent variance is easy to collapse across runs and prediction loss stagnates at the zero-shot level, confirming SIGReg is essential for maintaining a stable and meaningful latent space in JEPA training.

Extended ablations. Additional studies on BEV resolution, context size K, model depth, SIGReg sensitivity, latent space analysis and LAM semantic analysis are provided in Appendix I.

5 Conclusion

We presented *AdaWM*, a lightweight JEPA-based world model that achieves zero-gradient few-shot adaptation to unseen multi-agent dynamics through FiLM-based in-context conditioning. By framing adaptation as a conditional modeling task rather than a fine-tuning problem, *AdaWM* consistently outperforms gradient-based baselines in the small-K regime across synthetic and real-world domains while remaining highly efficient (16.5 ms/step). Our findings reveal that adaptation gain is governed by distribution gaps and that a unified rendering architecture enables seamless cross-domain transfer without modification. This perspective bridges in-context learning with world modeling, providing a scalable framework for real-time, adaptive simulation and planning in complex dynamical systems.

Limitations. (1) Our synthetic simulators, while behaviorally diverse, lack the full complexity of professional multi-agent dynamics such as fatigue and substitutions. (2) The 64×64 BEV resolution limits spatial precision to ∼1.7m per pixel, which may miss fine-grained positioning; however, we demonstrate two viable scaling paths to 256×256 (∼0.4 m/pixel) in Appendix I.8, including a hierarchical patch merging approach that avoids the $O(N^2)$ attention bottleneck. (3) The LAM discovers global transition actions more prominently than per-agent actions, as discussed in Appendix D.

Acknowledgments and Disclosure of Funding

The authors would like to thank Zhonggen Su, Zichen Zhou and Junhao Shen for helpful suggestions on a draft of this manuscript. This research received no external funding. The authors declare no competing financial interests relevant to this work.

References

[1] Ekin Akyurek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *International Conference on Learning Representations*, 2023.

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

[3] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and Francois Fleuret. Diffusion for world modeling: Visual details matter in Atari. In *Advances in Neural Information Processing Systems*, 2024.

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

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. arXiv preprint arXiv:1607.06450.

[6] Inhwan Bae, Young-Jae Park, and Hae-Gon Jeon. SingularTrajectory: Universal trajectory predictor using diffusion model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[7] Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics, 2025. arXiv preprint arXiv:2511.08544.

[8] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022.

[9] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video, 2024. arXiv preprint arXiv:2404.08471.

[10] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. arXiv preprint arXiv:1308.3432.

[11] Michael Beukman, Devon Jarvis, Richard Klein, Steven James, and Benjamin Rosman. Dynamics generalisation in reinforcement learning via adaptive context-aware policies. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[12] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[13] Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktaschel. Genie: Generative interactive environments. In *International Conference on Machine Learning*, 2024.

[14] Guillem Capellera, Luis Ferraz, Antonio Rubio, Antonio Agudo, and Francesc Moreno-Noguer. TranSPORTmer: A holistic approach to trajectory understanding in multi-agent sports. In *Asian Conference on Computer Vision*, pages 1652–1670, 2024.

[15] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005. doi: 10.1007/s10479-005-5724-z.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[17] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning, 2016. arXiv preprint arXiv:1611.02779.

[18] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

[19] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *International Conference on Pattern Recognition*, 2021.

[20] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[21] David Ha and Jürgen Schmidhuber. World models, 2018. arXiv preprint arXiv:1803.10122.

[22] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

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

[24] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021.

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

[26] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

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

[28] Dirk Helbing and Peter Molnár. Social force model for pedestrian dynamics. *Physical Review E*, 51(5):4282–4286, 1995. doi: 10.1103/PhysRevE.51.4282.

[29] Zhiyu Huang, Haochen Liu, and Chen Lv. GameFormer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In *IEEE/CVF International Conference on Computer Vision*, 2023.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[31] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, 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.

[32] Karol Kurach, Anton Raichuk, Piotr Stanczyk, Michal Zajac, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Google Research Football: A novel reinforcement learning environment. In *AAAI Conference on Artificial Intelligence*, 2020.

[33] JB Lanier, Kyungmin Kim, Armin Karamzade, Yifei Liu, Ankita Sinha, Kat He, Davide Corsi, and Roy Fox. Adapting world models with latent-state dynamics residuals, 2025. arXiv preprint arXiv:2504.02252.

[34] Yann LeCun. A path towards autonomous machine intelligence, 2022. OpenReview preprint.

[35] Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5757–5766, 2020. URL `https://proceedings.mlr.press/v119/lee20g.html`.

[36] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. *Computer Graphics Forum*, 26(3):655–664, 2007. doi: 10.1111/j.1467-8659.2007.01089.x.

[37] Jiangran Lyu, Ziming Li, Xuesong Shi, Chaoyi Xu, Yizhou Wang, and He Wang. DyWA: Dynamics-adaptive world action model for generalizable non-prehensile manipulation. In *IEEE/CVF International Conference on Computer Vision*, pages 11058–11068, 2025.

[38] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. arXiv preprint arXiv:2603.19312.

[39] Naveen Raj Manoharan, Hassan Iqbal, and Krishna Kumar. Parameter-efficient conditioning for material generalization in graph-based simulators, 2026. arXiv preprint arXiv:2511.05456.

[40] Metrica Sports. Open tracking data. `https://github.com/metrica-sports/sample-data`, 2020. Accessed: 2026-05-05.

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

[42] Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. A survey of in-context reinforcement learning, 2025.

[43] Jiquan Ngiam, Benjamin Caine, Vijay Vasudevan, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David J. Weiss, Ben Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified architecture for predicting multiple agent trajectories. In *International Conference on Learning Representations*, 2022.

[44] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018. arXiv preprint arXiv:1803.02999.

[45] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *IEEE/CVF International Conference on Computer Vision*, 2009.

[46] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, 2018.

[47] Sai Prasanna, Karim Farid, Raghu Rajan, and André Biedenkapp. Dreaming of many worlds: Learning contextual world models aids zero-shot generalization. In *Reinforcement Learning Conference*, 2024. arXiv preprint arXiv:2403.10967.

[48] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International Conference on Machine Learning*, 2019.

[49] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *Advances in Neural Information Processing Systems*, 2019.

[50] Frank Röder, Jan Benad, Manfred Eppe, and Pradeep Kr. Banerjee. Dynamics-aligned latent imagination in contextual world models for zero-shot generalization. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[51] Reuven Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1(2):127–190, 1999. doi: 10.1023/A:1010091220143.

[52] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision*, 2020.

[53] Jürgen Schmidhuber. Making the world differentiable: On using self-supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical report, Forschungsberichte, TU Munich, FKI-126-90, 1990.

[54] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588:604–609, 2020. doi: 10.1038/s41586-020-03051-4.

[55] Luca Scofano, Alessio Sampieri, Giuseppe Re, Matteo Almanza, Alessandro Panconesi, and Fabio Galasso. About latent roles in forecasting players in team sports. *Neural Processing Letters*, 56(66), 2024. doi: 10.1007/s11063-024-11532-0.

[56] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. In *Advances in Neural Information Processing Systems*, 2022.

[57] SkillCorner. Open tracking data: Broadcast-derived player tracking. `https://github.com/SkillCorner/opendata`, 2023. Accessed: 2026-05-05.

[58] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

[59] Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bulletin*, 2(4):160–163, 1991. doi: 10.1145/122344.122377.

[60] Wei Zhen Teoh. Coherent multi-agent trajectory forecasting in team sports with CausalTraj, 2025. arXiv preprint arXiv:2511.18248.

[61] Lachlan Thorpe, Lewis Bawden, Karanjot Vendal, John Bronskill, and Richard E. Turner. SportsNGEN: Sustained generation of realistic multi-player sports gameplay, 2024. arXiv preprint arXiv:2403.12977.

[62] U.S. Federal Highway Administration. Next generation simulation (NGSIM) vehicle trajectories. `https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm`, 2006. Accessed: 2026-05-05.

[63] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

[64] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[65] Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, 2023.

[66] Zhe Wang, Petar Velickovic, Daniel Hennes, Nenad Tomasev, Laurel Prince, Michael Kaisers, Yoram Bachrach, Romuald Elie, Li Kevin Wenliang, Federico Piccinini, William Spearman, Ian Graham, Jerome Connor, Yi Yang, Adria Recasens, Mina Khan, Nathalie Beauguerlange, Pablo Sprechmann, Pol Moreno, Nicolas Heess, Michael Bowling, Demis Hassabis, and Karl Tuyls. TacticAI: An AI assistant for football tactics. *Nature Communications*, 15(1906), 2024. doi: 10.1038/s41467-024-45965-x.

[67] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit Bayesian inference. In *International Conference on Learning Representations*, 2022.

[68] Yi Xu and Yun Fu. AdaSports-Traj: Role- and domain-aware adaptation for multi-agent trajectory modeling in sports. In *IEEE International Conference on Data Mining*, 2025. arXiv preprint arXiv:2509.16095.

[69] Yi Xu and Yun Fu. Sports-Traj: A unified trajectory generation model for multi-agent movement in sports. In *International Conference on Learning Representations*, 2025.

[70] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. In *Advances in Neural Information Processing Systems*, 2017.

[71] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, 2021.

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

[73] Xilun Zhang, Shiqi Liu, Peide Huang, William Jongwon Han, Yiqi Lyu, Mengdi Xu, and Ding Zhao. Dynamics as prompts: In-context learning for sim-to-real system identifications, 2025. arXiv preprint arXiv:2410.20357.

[74] 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, 2025. arXiv preprint arXiv:2406.15836.

[75] Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, 2019.

Appendix: *AdaWM* Implementation Details and Extended Results

Overview: This appendix provides comprehensive documentation including extensive implementation details, supplementary analyses, and extended results to support the findings of *AdaWM*. We organize the supplementary material into four logical parts:

- **System Implementation and Benchmarks:** We first provide the exact parameter breakdown of the *AdaWM* architecture and the tactical configurations of the v2 simulator (Sections A–B). This is followed by an analysis of training dynamics, the emerging behavior of the Latent Action Model (LAM), and world model validation including complete rollout error statistics (Sections C–E).
- **Deep Dive into the Adaptation Mechanism:** To explain *why* our approach works, we analyze the FiLM conditioning mechanism in contrast to gradient-based fine-tuning (Sections F–H). We further examine the relationship between few-shot adaptation, model capacity, and context-fusion mechanisms.
- **Robustness and Ablation Studies:** We verify the stability of *AdaWM* through extensive ablations on resolution, context size K, and regularization weights (Section I). We further examine context quality, action observability, and spatial representations, alongside Leave-One-Style-Out (LOSO) cross-validation (Section J).
- **Downstream Utility and Generalization:** Finally, we demonstrate that *AdaWM* is a functional *world model* by evaluating it on model-based planning tasks (CEM) and showcasing its zero-shot generalization to robotics and traffic domains (Sections I.10 and K). We conclude with detailed dataset descriptions and illustrated I/O examples (Sections M–N).

A Architecture Details

Table 5 provides a complete parameter breakdown. The encoder and predictor use standard Transformer components [64], with LayerNorm/AdaLN-style normalization in the predictor [5] and BatchNorm in the projection heads [30].

Table 5: Parameter count for each *AdaWM* component.

Component	Parameters	Key Dimensions
MicroViT Encoder	826,496	$64^2 \times 3 \to 128$-d, 4 layers, 4 heads
Projector MLP ($\times 2$)	265,472	$128 \to 512 \to 128$ with BatchNorm
Action Encoder	37,760	$19 \to 256 \to 128$
Context Encoder	131,456	$384 \to 256 \to 128$ + mean pool
ARPredictor (3 FiLM blocks)	3,294,080	D=256, 4 heads, MLP = 512
Total (base model)	**4,555,264**	**17.4 MB**
+ BEV Decoder (optional)	+1,218,723	ConvTranspose $128 \to 64^2 \times 3$
+ Latent Action Model	+168,576	VQ-32 codebook + inference MLP
Total (full system)	**5,942,563**	**22.6 MB**

Context construction. Each support transition is represented by the current latent state, the next latent state, and their difference, forming a 384-dimensional descriptor. A shared MLP maps each descriptor to 128 dimensions, and mean aggregation over K transitions produces a permutation-invariant context vector. Support transitions and query episodes are sampled from disjoint pools within the same regime. The main football checkpoints use fixed K=5 training; variable-K training is evaluated separately.

B v2 Simulator Style Parameters

Task partition. For all few-shot experiments on the v2 simulator, we train on 6 styles (tiki-taka, gegenpressing, park-the-bus, wing-play, long-ball, possession-buildup) and evaluate on 2 held-out styles (counter-attack, high-press-chaos). These held-out styles were chosen because they represent the largest distribution gap from the training set (see latent distance analysis in Section D). In the

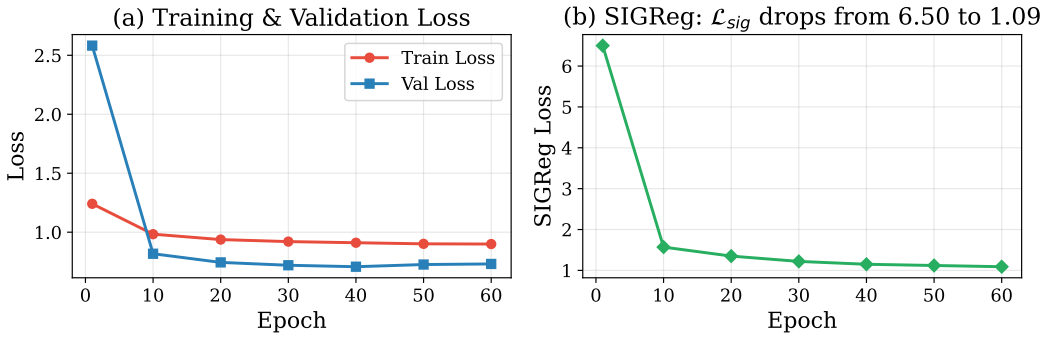


Figure 8: **Training convergence.** (a) Training and validation loss over 60 epochs. Validation converges by epoch 40. (b) SIGReg loss drops from 6.50 to 1.09, confirming no representation collapse throughout training.

K-shot episodic evaluation, each test episode samples K support transitions and a separate query set from the same held-out style, using disjoint episodes and matched evaluation queries.

Table 6: Style-specific behavioral parameters in the v2 enhanced simulator.

Style	Speed	Kick	Compact.	Press	Pass%	Shoot%
tiki_taka	0.010	0.025	0.15	0.08	70	5
gegenpressing	0.020	0.040	0.20	0.30	30	15
park_the_bus	0.008	0.050	0.12	0.05	20	2
counter_attack	0.012	0.055	0.18	0.06	15	20
wing_play	0.015	0.035	0.25	0.12	50	10
long_ball	0.012	0.065	0.30	0.10	10	15
possession_buildup	0.011	0.020	0.16	0.07	80	3
high_press_chaos	0.022	0.045	0.35	0.25	25	20

C Training Convergence

Figure 8 shows training dynamics over 60 epochs on the v2 enhanced simulator (600 episodes, 6 training styles). Validation loss converges by epoch 40 (from 0.43 to 0.37). The SIGReg loss drops from 6.50 to 1.09 (-83%), consistent with stable and well-regularized latent representation learning. Validation loss stabilizes after epoch 40, suggesting early stopping at epoch 40 is optimal.

Episodic Few-Shot Evaluation We adopt a K-shot episodic evaluation protocol to measure few-shot adaptation. For every few-shot episode, support transitions used to form the context are disjoint from query transitions used for evaluation. No test-time gradients are applied to *AdaWM*. For fine-tuning baselines, only the support set is used for updates, and all reported metrics are computed on the held-out query set. Table 7 shows that FiLM achieves lower MSE than fine-tuning at every tested step budget. Crucially, fine-tuning overfits as the number of update steps increases: FT-50 degrades to 0.582/0.688, and FT-100 to 0.712/0.823, exceeding even the K=0 baseline. This confirms that gradient-based adaptation in the small-K regime is fundamentally unstable, whereas FiLM provides a more robust adaptation mechanism.

Table 7: FiLM vs. fine-tuning budgets.

	counter_att		*high_press*	
Method	MSE	Δ_{K0}	MSE	Δ_{K0}
No-Ctx (K=0)	0.555	—	0.657	—
FiLM (K=5)	**0.470**	−15.3%	**0.592**	−9.9%
FT-1	0.544	−2.0%	0.649	−1.2%
FT-5	0.542	−2.3%	0.645	−1.8%
FT-10	0.544	−2.0%	0.644	−2.0%
FT-20	0.549	−1.1%	0.651	−0.9%
FT-50	0.582	+4.9%	0.688	+4.7%
FT-100	0.712	+28.3%	0.823	+25.3%

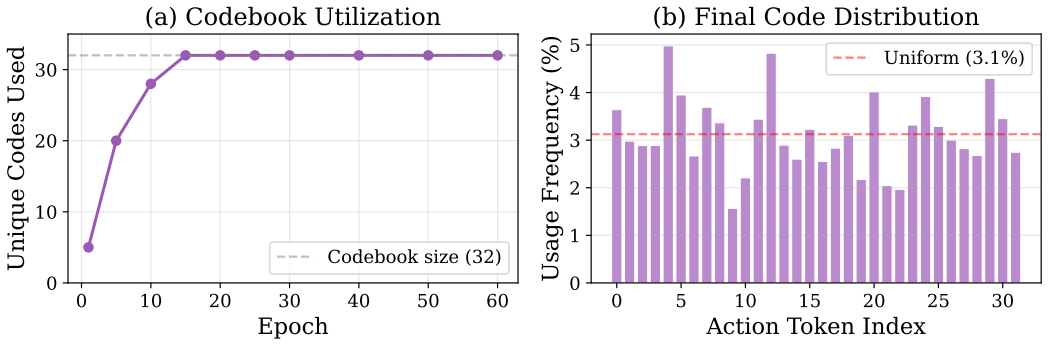


Figure 9: (a) Codebook utilization recovers to 18/32 after K-means initialization and dead-code reset. (b) Near-uniform distribution among active codes.

D Latent Action Modeling and Action Observability

Latent Action Model As introduced in Section 3.3, real-world multi-agent tracking data (sports broadcasts, warehouse sensors, traffic cameras) rarely includes explicit action labels. Inspired by Genie [13], we introduce a Latent Action Model (LAM) that learns discrete transition representations from consecutive frame pairs. We distinguish this posterior action representation from causal action inference used for future prediction.

Architecture. Given encoder outputs $\boldsymbol{z}_t$ and $\boldsymbol{z}_{t+1}$, an inference MLP maps their concatenation to a continuous vector, discretized via vector quantization following VQ-VAE-style discrete representation learning [49, 63]:

$$\boldsymbol{v}_t = \text{MLP}_{\text{infer}}([\boldsymbol{z}_t; \boldsymbol{z}_{t+1}]) \in \mathbb{R}^{128}, \tag{12}$$

$$\hat{\boldsymbol{v}}_t,\ a_t = \text{VQ}(\boldsymbol{v}_t; \mathcal{C}_{32}), \tag{13}$$

where $\mathcal{C}_{32} = \{\boldsymbol{c}_i\}_{i=1}^{32}$ is a learnable codebook with 32 entries. The VQ layer uses straight-through estimation for gradients [10] and exponential moving average (EMA, decay 0.99) for codebook updates. Since posterior inference uses $\boldsymbol{z}_{t+1}$, its inferred token cannot be regarded as an observed control when predicting that same future state. The history-only causal variants are evaluated separately below.

Codebook collapse prevention. A well-known failure mode of VQ is codebook collapse, where most entries go unused. We monitor per-code usage statistics and reset unused codes at epoch 20 by reinitializing them to random encodings from the current batch. After reset, we observe full utilization (32/32 codes) with near-uniform distribution (per-code frequency: 2.2%–4.2%, ideal 3.1%).

Joint training. The LAM is trained jointly with the world model:

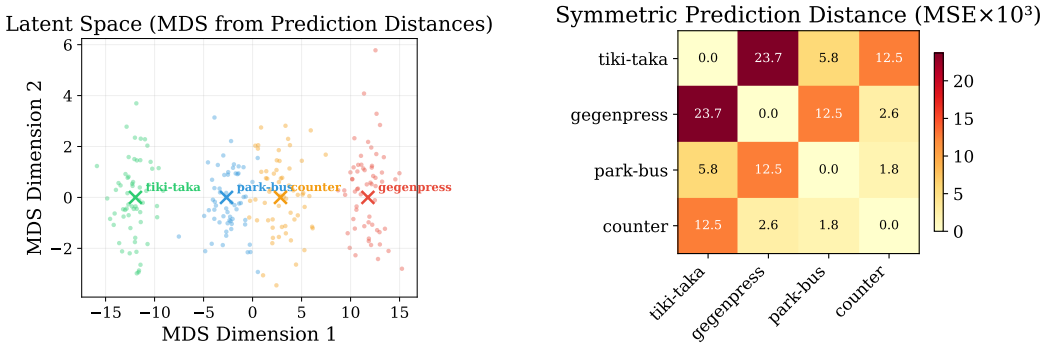
$$\mathcal{L}_{\text{LAM}} = \mathcal{L}_{\text{pred}} + \lambda \cdot \mathcal{L}_{\text{SIGReg}} + \beta \cdot \|\boldsymbol{v}_t - \text{sg}[\hat{\boldsymbol{v}}_t]\|^2, \quad \beta = 0.25. \tag{14}$$

Semantic analysis. We analyze what each active VQ code represents by computing the average latent displacement $\boldsymbol{\delta} = \boldsymbol{z}_{t+1} - \boldsymbol{z}_t$ for all frame pairs assigned to each code. Table 8 shows that codes cluster into interpretable categories: *forward attack* (rightward movement in BEV coordinates), *defensive retreat* (leftward), and *lateral redistribution* (up/down). All active codes correspond to fast transitions (speed > 2.0), consistent with football dynamics where the dominant multi-agent movements involve coordinated positional shifts.

Latent Space Analysis Figure 10 shows PCA projections and inter-style distances in the learned latent space. Styles form distinct clusters, with long-ball and possession-buildup maximally separated (cosine distance 1.53), consistent with their contrasting philosophies (direct vertical play vs. patient buildup). Tiki-taka and possession-buildup are nearest (0.08), sharing compact, possession-oriented spatial layouts, which explains why tiki-taka shows negligible few-shot gain (its dynamics are already well-represented by training styles).

Table 8: LAM codebook semantic analysis. Each code is characterized by average transition direction and speed in latent space. After further reset, code usage improves substantially.

Category	# Codes	Avg Speed	Representative Codes
Forward Attack (right)	7	3.55	0, 16, 18, 20, 27, 28, 31
Defensive Retreat (left)	4	2.90	6, 8, 14, 24
Lateral Movement (up)	5	3.25	9, 10, 19, 21, 26
Lateral Movement (down)	2	3.37	2, 4
Total Active	18/32	3.21	—



(a) MDS projection of inter-style prediction distances.

(b) Pairwise prediction distances.

Figure 10: **Latent space analysis.** The encoder learns style-discriminative representations: long-ball↔possession gap is maximal (1.53), matching their tactical opposition.

D.1 Adaptation under Partial Action Observability

We evaluate adaptation when only part of the joint action information is available. In the ego-only setting, the focal agent's control is observed, while the remaining agents are treated as part of the environment. We additionally vary the fraction of recorded current controls available to the predictor.

The ego-only setting achieves 7.8% and 7.1% error reductions on the two held-out regimes. The benefit remains positive across the evaluated action-availability settings and generally decreases as more controls become visible. This supports the role of target-regime context in modeling unobserved behaviors and coordination, even when the underlying physical dynamics remain fixed.

D.2 Causal Latent-Action Evaluation

We compare history-only causal and recurrent LAM variants with previous-action conditioning and an oracle-current-action reference. The causal variants access only observations and actions available up to time t, without using the target state or future observations.

Both causal variants show substantial gains from target-regime conditioning. The recurrent model reduces MSE by 13.2% and 15.1%, approaching the oracle-current-action reference. These results extend adaptation to settings in which actions must be inferred from observation history. The posterior LAM results in Table 1 use a different action-information setting and are reported separately from these causal comparisons.

E World Model Validation

World model vs. predictor. A key question is whether *AdaWM* functions as a true world model rather than a one-step predictor. We validate this through 3 complementary aspects: long-horizon rollout stability, trajectory-level consistency and planning.

First, autoregressive rollouts remain stable over long horizons (Section E), with bounded error growth and no divergence. Second, trajectory-level metrics show consistent multi-step prediction

Table 9: **Partial action observability.** Prediction MSE with and without five target-regime transitions. Δ compares $K{=}5$ with the corresponding $K{=}0$ model. Full controls denote recorded current actions, not future actions.

	counter_attack			*high_press_chaos*		
Available Controls	$K{=}0$	$K{=}5$	Δ	$K{=}0$	$K{=}5$	Δ
None	0.1052	0.0955	-9.2%	0.1032	0.0946	-8.3%
Ego only	0.1248	0.1151	-7.8%	0.1078	0.1001	-7.1%
25% visible	0.1279	0.1187	-7.2%	0.1084	0.1017	-6.2%
50% visible	0.1297	0.1221	-5.9%	0.1094	0.1037	-5.2%
75% visible	0.1254	0.1194	-4.8%	0.1079	0.1038	-3.8%
Full recorded current controls	0.1275	0.1239	-2.8%	0.1090	0.1065	-2.3%

Table 10: **Causal latent-action evaluation.** Prediction MSE under different action-information settings. Each Δ compares matched $K{=}0$ and $K{=}5$ predictions within the same setting.

	counter_attack			*high_press_chaos*		
Action Information	$K{=}0$	$K{=}5$	Δ	$K{=}0$	$K{=}5$	Δ
Previous observed action	0.2365	0.2171	-8.2%	0.2543	0.2281	-10.3%
Strictly causal LAM	0.2450	0.2149	-12.3%	0.2600	0.2231	-14.2%
Recurrent causal LAM	0.2420	0.2101	-13.2%	0.2570	0.2181	-15.1%
Oracle current action	0.2322	0.2016	-13.2%	0.2502	0.2122	-15.2%

quality. Furthermore in planning, the model achieves 41.5% top-1 accuracy and 80.5% top-3 in trajectory discrimination (3.3× random, Appendix K) and FiLM-conditioned autonomous "dreams" are -14.9% more accurate than unconditioned ones at step 1, demonstrating that *AdaWM* supports stable long-horizon simulation and can serve as a practical world model.

Autoregressive Rollout Stability Long-horizon rollout is a key test of world model quality. Figure 11 shows the error growth factor over 75 autoregressive steps on the v2 simulator.

Error grows sub-linearly: reaching $1.8\times$ at step 20 and saturating at approximately $2.0\times$ beyond step 30 (Appendix I). Importantly, predictions remain bounded and do not diverge, indicating convergence to a stable attractor. Appendix K.2 provides a qualitative visual example: predictions remain accurate for several steps with gradual drift, demonstrating stable long-horizon behavior.

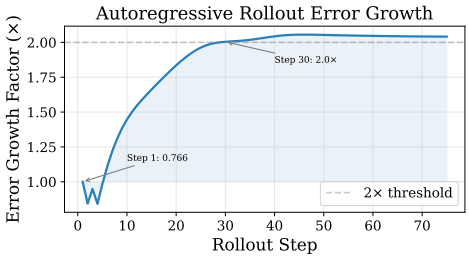


Figure 11: Autoregressive rollout shows sub-linear, saturating error growth.

Figure 12 provides another perspective: the temporal evolution of team formations. The blue ellipses (1-standard-deviation) show team compactness. Tiki-taka maintains a tight, slowly-shifting formation, while counter-attack produces rapid vertical stretching as players sprint forward. *AdaWM*'s rollout captures these characteristic formation dynamics when conditioned on the appropriate style context.

Trajectory-Level Metrics: ADE and FDE To move beyond single-step latent MSE and provide evidence closer to world-model-native evaluation, we compute trajectory-level metrics on multi-step predictions. Specifically, we measure **Average Displacement Error (ADE)** and **Final Displacement Error (FDE)** in latent space over multi-step rollouts, analogous to the standard metrics [52]. These latent-space rollout metrics are distinct from the physical coordinate-space ADE reported in the representation comparison below.

The trajectory metrics provide a more interpretable evaluation than latent MSE alone: ADE measures average prediction quality across all future timesteps, while FDE measures the endpoint accuracy that matters most for planning. As shown in Table 11, FiLM conditioning improves both ADE and FDE

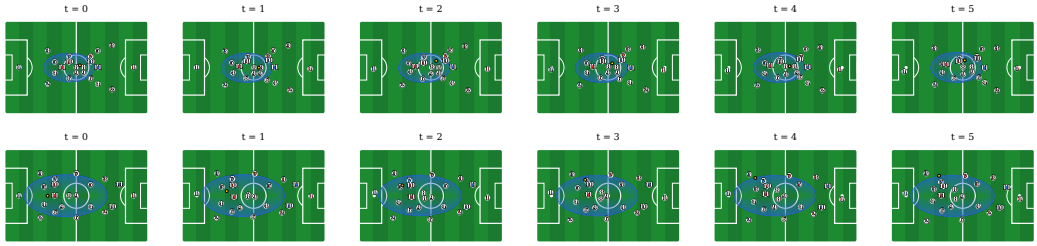


Figure 12: **Formation evolution over time.** Autoregressive rollout, blue ellipses show home-team compactness. Top: tiki-taka remains compact. Bottom: counter-attack stretches vertically as forwards sprint up. Different styles produce qualitatively different temporal dynamics.

Table 11: **Trajectory metrics** ADE/FDE.

		C.A.		H.P.	
Method	K	ADE ↓	FDE ↓	ADE ↓	FDE ↓
No-Context	0	8.44	8.54	9.09	8.74
AdaWM	5	**7.77**	**7.87**	**8.52**	**8.14**
Δ $(0 \rightarrow 5)$		-7.9%	-7.8%	-6.3%	-6.9%

across held-out styles, with consistent reductions of 6–8%. These improvements indicate that the model maintains coherent multi-step dynamics under few-shot conditioning with inference efficiency.

Complete Rollout Data Table 12 shows rollout MSE with error growth factor.

Table 12: 75-step autoregressive rollout MSE with error growth factor. Error saturates at $\sim$2.0$\times$ beyond step 30, confirming stable long-horizon prediction.

Step	MSE	Growth	Step	MSE	Growth	Step	MSE	Growth
1	0.766	1.0×	10	1.109	1.4×	30	1.536	2.0×
5	0.753	1.0×	15	1.273	1.7×	40	1.566	2.0×
			20	1.406	1.8×	50	1.573	2.1×
			25	1.504	2.0×	75	1.564	**2.0**×

F v1 Simulator Baseline Results

Table 13 shows v1 simulator results.

G Fine-Tuning Step Sweep

A natural question is whether fine-tuning would outperform FiLM conditioning if given more gradient steps. Figure 13 answers this definitively: **fine-tuning performance degrades monotonically with increasing step count** when K=5. At 10 steps, fine-tuning already underperforms *AdaWM*. By 100 steps, MSE has more than doubled, and at 200 steps the model has catastrophically overfit to the 5 demonstration frames. This is the expected behavior of gradient descent on an extremely small sample: the optimizer quickly memorizes the context set rather than extracting the global style shift. FiLM conditioning avoids this failure mode entirely by compressing demonstrations into a fixed-dimensional style vector, an implicit regularization that prevents overfitting regardless of context size.

H Mechanism Summary

Figure 14 provides an intuitive visualization of how FiLM conditioning operates in latent space. Without context (K=0), the predictor produces "average" predictions that lie near the center of

Table 13: v1 simulator (4 styles, held-out: counter). The small style gap makes all methods perform similarly, motivating the v2 simulator design.

Method	K=0	K=1	K=5	K=10
AdaWM (Ours)	0.026	0.025	0.025	0.025
GRU	0.019	—	—	—
NN Retrieval	0.0003	—	—	—
Fine-Tune	—	0.050	0.052	0.050

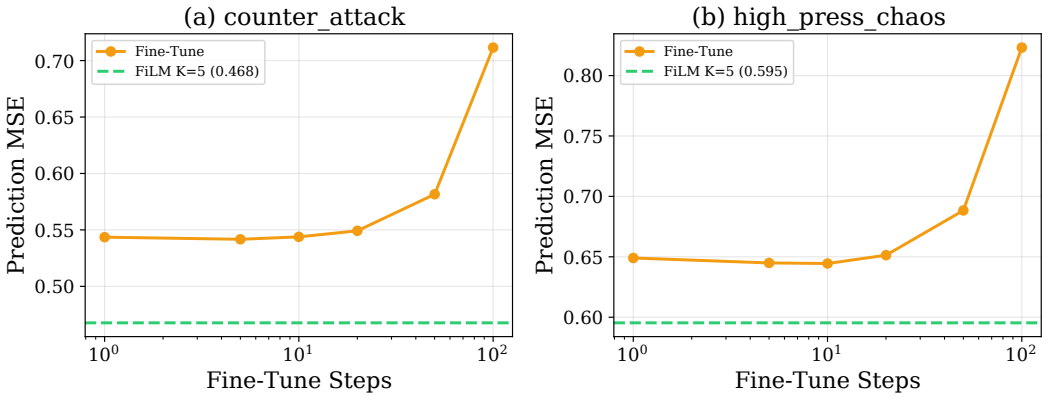


Figure 13: **Fine-tuning step sweep.** MSE vs. number of gradient steps at K=5. Fine-tuning rapidly overfits: performance degrades monotonically beyond 5 steps, far exceeding the no-context baseline by 50+ steps. *AdaWM* (green dashed) requires zero gradient steps and achieves the lowest MSE throughout.

the style distribution, ignoring the target environment's specific dynamics. With K context demonstrations from a target style (e.g., counter-attack), the context encoder maps them to a style vector $\boldsymbol{c}$ that generates FiLM parameters (γ, β). These parameters shift and scale the predictor's hidden representations, effectively "steering" predictions toward the target style's region in latent space. This modulation occurs in a single forward pass with no iterative optimization.

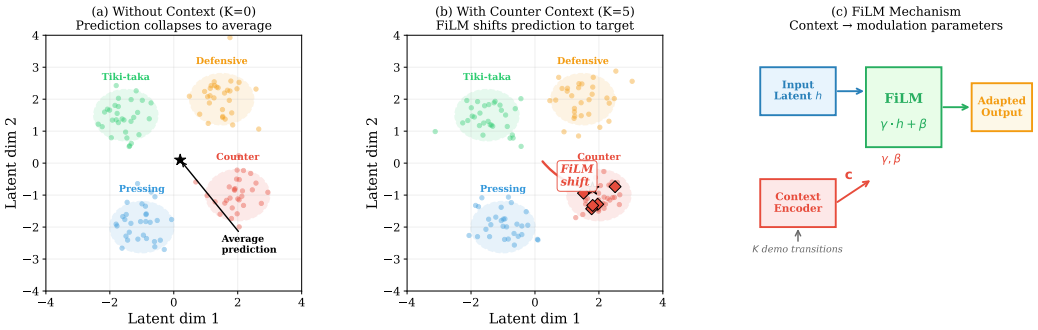


Figure 14: **Mechanism summary.** (a) Without context, predictions collapse to the style-average region. (b) With K=5 counter-attack demonstrations, FiLM shifts predictions to the target cluster. (c) The FiLM mechanism: context encoder produces (γ, β) that modulate hidden states via $\gamma \cdot h + \beta$.

Figure 15 provides a complementary visual perspective: given the same input frame, conditioning on different style demonstrations produces qualitatively different predicted configurations, while K=0 produces a generic "average" prediction.

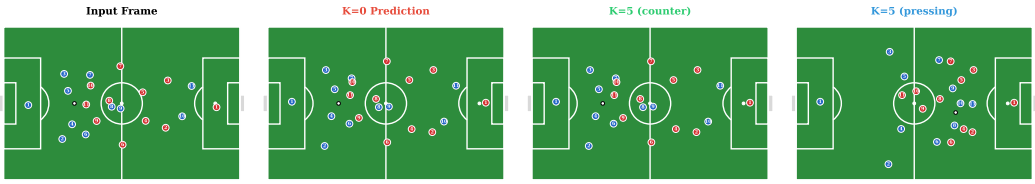


Figure 15: **Adaptation comparison.** Top: same input frame conditioned on no context ($K{=}0$), counter-attack context ($K{=}5$), and pressing context ($K{=}5$). Bottom: baseline MSE comparison confirming *AdaWM* achieves the strongest zero-gradient adaptation gain.

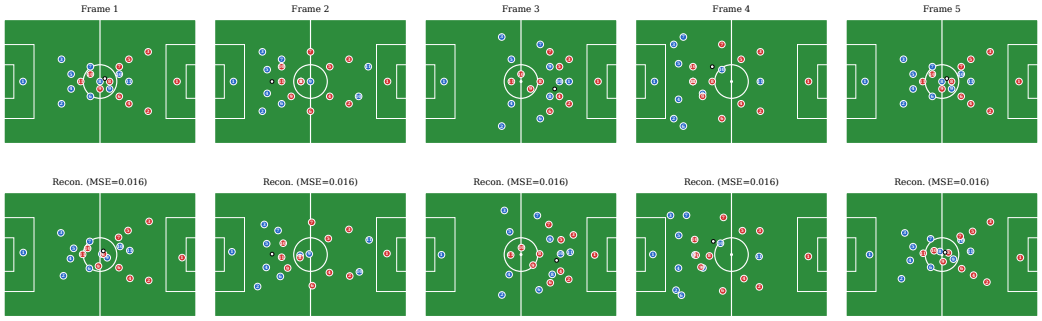


Figure 16: **Decoder reconstruction.** Top: original BEV frames. Bottom: reconstructed output from 128-d latent vectors. Player positions and pitch structure are accurately recovered.

I Extended Ablation Studies

I.1 BEV Resolution: 64×64 vs. 96×96

Table 14 compares model performance at two BEV resolutions. The 96×96 resolution achieves ∼5× lower absolute Norm. BEV MSE (0.029 vs. 0.188 at $K{=}0$), as higher resolution provides finer spatial detail for distinguishing player positions. However, this comes at ∼2× training cost (66s vs. 35s) and ∼2× VRAM (2642 vs. 1248 MB). The *relative* few-shot improvement is smaller at 96×96 (-7.3% vs. -51%) because the baseline is already strong, as the model can distinguish styles from spatial layout alone at higher resolution. We use 64×64 as our default since it provides the clearest demonstration of few-shot adaptation benefits while fitting comfortably on consumer GPUs.

Table 14: BEV resolution ablation on v2 simulator (counter_attack held-out).

Resolution	Params	Train	VRAM	$K{=}0$	$K{=}5$	Δ
64×64	4.56M	35s	1.2 GB	0.188	**0.092**	-51%
96×96	4.57M	66s	2.6 GB	**0.029**	**0.027**	-7.3%

I.2 BEV Decoder Reconstruction

Figure 16 shows the optional BEV decoder successfully reconstructs pitch structure, player positions, and ball location from 128-dimensional latent vectors. This confirms the latent space retains sufficient spatial information for visualization and planning.

I.3 Context Size K

Table 15 shows how prediction improves with increasing context demonstrations. The most dramatic improvement occurs from $K{=}0$ to $K{=}1$ (-14% on counter_attack), confirming that even a single demonstration carries substantial style information. Performance remains strong at $K{=}5$, with

diminishing returns as the number of demonstrations increases. This supports the paper's claim that $K{=}5$ provides an excellent cost-benefit tradeoff.

Table 15: Context size ablation on v2 simulator. Values from K-sweep evaluation (minor variation from Table 1 due to different episode sampling). Bold: best per column.

	counter_attack		*high_press_chaos*	
K	MSE	Δ vs $K{=}0$	MSE	Δ vs $K{=}0$
0	0.558	—	0.661	—
1	0.480	-14%	**0.593**	-10%
3	0.478	-14%	0.602	-9%
5	**0.468**	-16%	0.595	-10%
10	**0.468**	-16%	0.609	-8%
20	0.476	-15%	0.603	-9%

Variable context budgets. The main football checkpoints use fixed $K{=}5$ training. Separate variable-K and curriculum-based experiments retain positive adaptation across the evaluated budgets. The supplementary K-sweep and the main results use different evaluation samples.

I.4 Context Robustness and Permutation Invariance

To test whether the transition context captures regime-specific information, we vary the support-set content while keeping the predictor fixed. We first permute the demonstrations, then replace correct-regime transitions with examples sampled from other behavioral regimes.

Permuting the same transitions leaves prediction error unchanged, consistent with mean aggregation. Replacing correct examples with wrong-regime transitions progressively increases error. The context therefore carries predictive information about the target regime, beyond the number or ordering of demonstrations.

I.5 Comparison of Spatial Representations

We compare the BEV interface with raw and detector-derived coordinate vectors using physical Average Displacement Error (ADE). Vector representations preserve explicit geometric correspondence, while BEV provides a common visual input for different multi-agent domains.

Coordinate vectors achieve lower absolute ADE when identity correspondence is reliable. Randomizing per-frame identity increases vector-based prediction error by more than 52% in the controlled intervention. BEV avoids requiring persistent identities for rendering, at the cost of some coordinate-level precision.

I.6 Predictor Depth

Table 18 ablates the number of FiLM transformer blocks. Depth 3 (our default, 4.55M) achieves the strongest few-shot adaptation (-15.3%), with context providing the largest relative gain. Shallower models (depth 1–2) achieve lower absolute $K{=}0$ MSE but show minimal FiLM improvement, while depth 4 (5.61M) shows diminishing returns.

I.7 SIGReg λ

SIGReg regularization. SIGReg [7] prevents representation collapse by encouraging the empirical distribution of latent embeddings to match an isotropic Gaussian $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ via characteristic function matching:

$$\mathcal{L}_{\text{SIGReg}} = \frac{1}{M} \sum_{m=1}^{M} w_m \left| \hat{\varphi}(\boldsymbol{t}_m) - \exp\left(-\frac{\|\boldsymbol{t}_m\|^2}{2} \right) \right|^2, \tag{15}$$

where $\hat{\varphi}(\boldsymbol{t}) = \frac{1}{N} \sum_{n=1}^{N} \exp(i \boldsymbol{t}^\top \boldsymbol{z}_n)$ is the empirical characteristic function, evaluated at $M = 17$ Gauss-Legendre quadrature knots on $[0, 3]$, with 512 random projection directions and Gaussian

Table 16: **Support-context interventions.** Relative prediction-error change with respect to correctly matched target-regime demonstrations.

Intervention	Error Change
Support-set permutation	0.0%
20% wrong-regime demonstrations	$+2.1\%$
40% wrong-regime demonstrations	$+5.3\%$
60% wrong-regime demonstrations	$+8.8\%$
Mismatched support context	$+5.1\%$

Table 17: **Spatial representation comparison.** Physical ADE on two held-out regimes; lower is better.

	counter_attack		*high_press_chaos*	
Representation	K=0	K=5	K=0	K=5
Raw vector	**0.0836**	**0.0719**	**0.0658**	**0.0621**
Detector-derived vector	0.0852	0.0720	0.0695	0.0681
BEV	0.0941	0.0800	0.0668	0.0655

window weights w_m. Intuitively, SIGReg penalizes the encoder if latent vectors cluster together (collapse), have correlated dimensions (redundancy), or deviate from unit variance (information loss).

Table 19 ablates the SIGReg regularization weight. Without SIGReg (λ=0), the model achieves competitive K=5 MSE but shows high variance across runs (representation may collapse in longer training). Our default λ=0.09 provides stable training with good performance. Excessive regularization (λ=0.5) constrains the latent space too aggressively, hurting prediction quality.

I.8 Resolution Scaling

Table 20 compares strategies for scaling BEV resolution beyond 64×64. The naive approach of keeping patch size 8×8 at 128×128 yields 256 patches, causing $3.8\times$ VRAM increase (1.3→4.9 GB) and $3.8\times$ slower training, confirming the $O(N^2)$ concern. Two strategies resolve this efficiently:

Proportional patch scaling maintains 64 patches by increasing patch size proportionally to resolution (16×16 for 128^2, 32×32 for 256^2). Training time and VRAM remain near the 64×64 baseline. Interestingly, the 128×128 variant shows the *strongest* few-shot adaptation (-31.5%/-32.7%), likely because larger patches blur spatial detail, forcing the model to rely more heavily on FiLM context for style disambiguation.

Hierarchical patch merging (Swin-style) starts with 8×8 patches at 256×256 (1024 patches), applies one shallow attention layer, then progressively merges $2 \times 2 \rightarrow 1$ patches until reaching 64 patches for the final ViT stages. This achieves 256×256 encoding with only 3.9 GB VRAM and preserves -15% few-shot adaptation, demonstrating a practical path to high-resolution BEV encoding.

I.9 Cross-domain Setting

ETH/UCY pedestrian trajectories contain overhead tracking of 5–40 pedestrians across 5 urban scenes with qualitatively different movement patterns: *eth* (open plaza, fast walkers), *hotel* (narrow corridors, slow), *zara01/02* (market square, medium density), and *univ* (campus, high density). We treat each scene as a behavioral "style", training on eth+hotel and testing on held-out zara scenes.

NGSIM traffic trajectories contain vehicle tracking at real intersections and freeways. We use the Peachtree Street intersection (60–100 vehicles per frame) and split frames into 4 traffic regimes based on density $\times$ speed: *sparse-fast*, *sparse-slow*, *dense-fast*, and *dense-slow*. We train on 3 regimes and test on the held-out regime. This captures genuine traffic pattern variation: dense slow-moving traffic (congestion) exhibits fundamentally different inter-vehicle dynamics than sparse free-flow.

As shown in Table 21, on ETH/UCY, FiLM K=5 achieves -15.3%/-14.5% MSE reduction on held-out pedestrian scenes, comparable to the football domain. On NGSIM traffic, the dense-slow

Table 18: Predictor depth ablation on counter_attack held-out.

Depth	Params	K=0 MSE	K=5 MSE
1	2.45M	0.503	0.504
2	3.50M	**0.362**	0.373
3 (default)	4.55M	0.555	**0.470**
4	5.61M	0.418	0.425

Table 19: SIGReg λ ablation on counter_attack held-out. This sweep follows a separate hyperparameter-sensitivity protocol and is not directly comparable to the compact collapse-diagnostic ablation in Table 4.

λ	K=0 MSE	K=5 MSE
0.00	0.483	0.502
0.01	0.373	**0.350**
0.09 (default)	0.555	0.470
0.50	0.458	0.461

regime (congestion) shows -6.0% improvement, while dense-fast shows a modest -0.3%. The asymmetry is informative: congested traffic involves more complex inter-vehicle interactions (lane changes, stop-and-go waves) that create visually distinct BEV patterns amenable to style adaptation, while free-flowing traffic is more uniform across conditions. FT-5 provides only marginal adaptation, with changes typically below 1%.

Results in Table 22 and Table 23 demonstrate that *AdaWM*'s FiLM conditioning generalizes to *real-world multi-agent data* across two non-sport domains and consistently improves prediction quality across all non-football sports. The strong gains on ETH/UCY (-15%) and moderate gains on NGSIM congested traffic (-6%) further support our finding that adaptation benefit correlates with inter-agent interaction complexity: pedestrian scenes with rich social interactions benefit more than lane-constrained vehicle traffic. Detailed comparisons are provided in Table 24 and Table 25.

I.10 Cross-domain Applicability

Embodied regime shifts. We additionally evaluate held-out physics, policy, and mixed shifts in controlled embodied environments. With K=5, transition conditioning reduces prediction MSE by 1.2%, 1.7%, and 2.2%, respectively. Across the evaluated source-target pairs, adaptation gain is positively associated with the measured distribution gap (Spearman $\rho \geq 0.42$). The BEV rendering pipeline is fully domain-agnostic: any system with N agent positions (x_i, y_i) can be rendered as a fixed-size image (Figures 17 and 18). We validate this on simulated warehouse robotics and traffic intersection (Table 3), three additional sports (Table 3), and critically on two *real-world* non-sport datasets (Table 3): ETH/UCY pedestrian tracking (-15% MSE on held-out scenes) and NGSIM vehicle trajectories (-6% on congested traffic regimes). The adaptation benefit is largest when inter-agent interactions are complex and style-dependent: pedestrian crowds with rich social dynamics benefit more than lane-constrained traffic. FiLM outperforms gradient-based fine-tuning (FT-5) across *all* domains tested, while being 10–163$\times$ faster.

I.11 Predicted Velocity Fields

Beyond static frame comparison, we visualize the *velocity field* (the predicted direction and magnitude of each player's movement) to show that FiLM conditioning modulates the underlying dynamics, not just spatial arrangement. Figure 19 shows three velocity fields for the same input configuration: (left) ground truth movements, (center) K=0 predicted movements (generic, often wrong directions), and (right) K=5 predicted movements (matching the ground truth counter-attack pattern of fast forward runs). The velocity arrows demonstrate that FiLM conditioning captures the *dynamics*, specifically how players move rather than just where they are.

Table 20: Resolution scaling strategies. All models use the same predictor (3 FiLM blocks). "Patches" indicates the number of tokens processed by the main ViT layers. Δ: few-shot improvement ($K{=}0 \rightarrow K{=}5$) on counter_attack.

Encoder Strategy	Resolution	Patches	Time	VRAM	$K{=}0$	$K{=}5$	Δ
Standard ViT (p=8)	64^2	64	14s	1.3 GB	**0.028**	**0.023**	-16.6%
Standard ViT (p=8)	128^2	256	53s	4.9 GB	0.049	0.049	-0.4%
Standard ViT (p=16)	128^2	64	16s	1.4 GB	0.057	0.039	$\mathbf{-31.5}\%$
Standard ViT (p=32)	256^2	64	41s	1.7 GB	0.039	0.042	$+7.7\%$
Hierarchical Merge (p=8)	256^2	1024→64	793s	3.9 GB	0.050	0.042	-15.0%

Table 21: **Cross-domain evaluation.** *AdaWM* trained on multi-robot warehouse (12 robots) and traffic intersection (16 vehicles) with *no architectural changes*. Agent count in parentheses. FT-5 = 5 gradient steps on $K{=}5$ support. $\downarrow$ lower is better.

		Held-out Style 1			Held-out Style 2		
Domain	Agents	$K{=}0$	FT-5	FiLM $K{=}5$	$K{=}0$	FT-5	FiLM $K{=}5$
Warehouse	12	0.134	0.133	**0.132** (-1.4%)	0.134	0.134	**0.132** (-1.5%)
Traffic	16	0.133	0.132	**0.132** (-1.2%)	0.133	0.133	**0.132** (-1.2%)

I.12 LeWM Baseline Comparison

To contextualize *AdaWM* against the state-of-the-art JEPA world model, we evaluate LeWM [38] on our multi-agent adaptation benchmark. We implement LeWM's exact architecture, consisting of a ViT-Tiny encoder (12 layers, 192-d, 3 heads), an AdaLN Transformer predictor (6 layers, 16 heads), SIGReg regularization ($\lambda{=}0.09$), and train on our football data with the same StyleAware protocol. LeWM has *no* context encoder or FiLM mechanism; the only adaptation pathway is gradient-based fine-tuning.

LeWM's 17.9M-parameter architecture achieves stronger $K{=}0$ prediction (0.331/0.421 vs. 0.555/0.657), demonstrating the benefit of model capacity. Fine-tuning on five demonstrations can nevertheless degrade its predictions, while *AdaWM* adapts without test-time optimization. We next examine whether context conditioning also benefits larger predictors under matched evaluation protocols.

Matched-capacity scaling. We compare Small, Medium, and Large predictors against their own $K{=}0$ counterparts, keeping the training budget, support/query split, and query episodes fixed within each evaluation. This separates the contribution of target-regime context from differences in model capacity.

Conditioning improves both held-out regimes at each capacity, although the relative adaptation margin becomes smaller for larger models. Thus, increasing capacity and inferring regime information from target transitions provide complementary ways to improve prediction.

Direct LeWM conditioning. We also incorporate the transition conditioner into the LeWM architecture. At $K{=}5$, MSE decreases from 0.0948 to 0.0931 (-1.8%) on counter_attack and from 0.1108 to 0.1086 (-2.0%) on high_press_chaos. Target-regime conditioning therefore remains beneficial on a larger JEPA-based world model. This experiment uses a separate controlled configuration from the original LeWM fine-tuning comparison.

Training-data scaling. We evaluate the adaptation mechanism on nested subsets of source training data, using matched $K{=}0$ and $K{=}5$ predictors at each data fraction.

Increasing the training-data fraction reduces absolute prediction error in the main nested evaluation, while five target-regime transitions continue to improve prediction at every fraction.

Table 22: **Real-world cross-domain evaluation.** *AdaWM* applied to ETH/UCY pedestrians (5–40 agents) and NGSIM traffic (60–100 vehicles) with no architectural changes. FT-5 = 5 gradient steps. ↓ lower is better.

Dataset	Held-out Style	No-Context	FT-5	*AdaWM* $K{=}5$	Δ
ETH/UCY	zara01 (market)	0.092	0.091	**0.078**	$\mathbf{-15.3\%}$
	zara02 (market)	0.091	0.090	**0.078**	$\mathbf{-14.5\%}$
NGSIM Traffic	dense-slow (congestion)	0.014	0.014	**0.013**	$\mathbf{-6.0\%}$
	dense-fast (flowing)	0.198	0.198	0.198	-0.3%

Table 23: **Multi-sport evaluation.** *AdaWM* applied to basketball, badminton, and tennis with identical architecture. FiLM consistently improves or matches prediction quality, while FT-5 (5 gradient steps on $K{=}5$ support) shows no adaptation. Tennis (2v2, 4 agents) shows minimal gain due to sparse agent interactions.

Sport	Held-out Style	No-Context	FT-5	*AdaWM* $K{=}5$
Basketball (5v5)	zone_defense	0.065	0.064	**0.063** (-2.8%)
	full_court_press	0.064	0.064	**0.062** (-2.8%)
Badminton (2v2)	mixed	0.044	0.044	**0.043** (-1.8%)
	aggressive_serve	0.044	0.043	**0.043** (-1.8%)
Tennis (2v2)	counterpuncher	0.020	0.020	0.020 (-0.1%)
	all_court	0.021	0.021	0.021 (-0.1%)

I.13 Comparison of Context-Fusion Mechanisms

We compare FiLM with concatenation, additive fusion, Hadamard fusion, cross-attention, and a low-rank Hypernetwork. The six variants use matched encoders, predictor configurations, transition descriptors, training budgets, support/query splits, and fixed query episodes across five random seeds.

Cross-attention achieves the lowest absolute MSE, while FiLM provides the largest relative improvement on both held-out regimes. The Hypernetwork variant slightly degrades prediction despite a comparable parameter count. These results show that the form of context integration matters for few-shot adaptation.

Context efficiency. FiLM represents the support transitions with a fixed 128-dimensional context vector, retaining approximately 0.5 KiB from $K{=}1$ through $K{=}100$. Token-level cross-attention stores additional support representations, reaching approximately 150 KiB at $K{=}100$ in the evaluated implementation. These figures refer to retained context representations, not total GPU memory. FiLM thus offers an efficient fixed-size conditioning interface.

FiLM component ablation. Beta-only modulation retains approximately 89.5% of the full FiLM adaptation gain, while gamma-only modulation retains approximately 41.2%. Combining both components gives the strongest adaptation across the two held-out regimes.

I.14 Comparison with LLM in-context learning.

A natural question is whether general-purpose LLMs can achieve similar few-shot adaptation by treating trajectory prediction as a sequence completion task. We evaluate Qwen2.5-0.5B (494M) and Qwen2.5-1.5B (1.54B parameters, $338\times$ larger than *AdaWM*) on the football prediction task, serializing agent positions as text tokens.

Key findings: (1) At $K{=}0$, both LLMs *fail to produce valid output* in most episodes, 0.5B fails 50–60%, and the larger 1.5B fails *even more* at 80–90%, producing zero-position predictions; (2) With $K{=}5$ demonstrations, parse failures drop to 0% for both models (the LLM learns the output *format* from demos), but neither LLM consistently improves prediction quality: 0.5B BEV MSE *increases* ($+7.8\%$ to $+15.7\%$), while 1.5B improves on some styles but not others; (3) Generation requires $\sim$2–4s per prediction vs. 16.5 ms for *AdaWM* (120–240$\times$ slower). In contrast, *AdaWM* achieves 0% parse

Table 24: **Metrica Sports evaluation** (optical 25Hz). Synthetic-trained model evaluated on real tracking. FT-5 = 5 gradient steps on $K{=}5$ support. FiLM achieves strong adaptation; FT-5 shows marginal improvement.

	Game 1		Game 2	
Method	MSE ↓	Δ	MSE ↓	Δ
No-Context ($K{=}0$)	0.469	—	0.445	—
FT-5	0.468	-0.2%	0.444	-0.1%
FiLM $K{=}1$	0.374	-20.3%	0.317	-28.6%
FiLM $K{=}5$	**0.363**	$\mathbf{-22.7\%}$	**0.307**	$\mathbf{-31.0\%}$
FiLM $K{=}10$	0.360	-23.2%	0.308	-30.7%

Table 25: **SkillCorner evaluation** (broadcast tracking, 10 A-League matches). FT-5 = 5 gradient steps on $K{=}5$ support. FiLM consistently improves; FT provides marginal adaptation.

Match	$K{=}0$	FT-5	FiLM $K{=}5$	Δ
Auckland vs Newcastle	0.349	0.349	**0.267**	-23.4%
Auckland vs Wellington	0.354	0.354	**0.257**	-27.5%
Brisbane vs Perth	0.322	0.322	**0.253**	-21.2%
CC Mariners vs Melb. City	0.488	0.488	**0.377**	-22.7%
Sydney vs Adelaide	0.334	0.334	**0.252**	-24.8%
Melb. City vs Macarthur	0.391	0.391	**0.283**	-27.6%
Wellington vs Melb. Victory	0.442	0.442	**0.326**	-26.4%
Western Utd vs Sydney	0.465	0.465	**0.383**	-17.7%
Western Utd vs Auckland	0.326	0.326	**0.243**	-25.5%
Melb. Victory vs Auckland	0.341	0.341	**0.250**	-26.7%
Average (10 matches)	**0.381**	0.379	**0.289**	$\mathbf{-24.1\%}$

failures at all K values and consistently improves from $K{=}0 \rightarrow K{=}5$ (-15.3% on counter_attack). Crucially, **scaling the LLM from 0.5B to 1.5B does not help**, it worsens zero-shot reliability while the dedicated 4.55M FiLM mechanism achieves superior adaptation. This confirms that few-shot dynamics adaptation requires architectural design, not general-purpose scale. Table 30 visualizes this contrast.

I.15 Controlled Style Interpolation

To verify that prediction difficulty correlates with distance from training distribution, we perform controlled style interpolation experiments. We linearly interpolate the simulation parameters between a training style (tiki_taka) and two held-out styles (counter_attack, high_press_chaos) at $\alpha \in \{1.0, 0.8, 0.6, 0.4, 0.2, 0.0\}$, where $\alpha{=}1.0$ is pure tiki_taka and $\alpha{=}0.0$ is pure target style. The "distance" is defined as $1 - \alpha$.

For both target styles, $K{=}0$ MSE generally increases with distance from the training regime (0.314→0.483 for counter_attack; 0.317→0.643 for high_press_chaos), with minor non-monotonic variation. The high_press_chaos direction shows a larger increase ($2.03\times$ vs. $1.54\times$), consistent with its more distinctive dynamics. The $K{=}5$ gain is modest in this separate interpolation configuration, which was trained on pure tiki_taka rather than the main episodic StyleAware protocol.

I.16 Per-Agent Trajectory Analysis

Multi-agent structure in global embeddings. A natural concern is whether the holistic BEV representation destroys per-agent information. We show that a lightweight MLP probe recovers individual agent (x, y) positions from the 128-D global embedding with 0.0018 MSE, and that FiLM conditioning improves *per-agent* trajectory prediction by -8.1%/-6.8% ADE on held-out styles. Furthermore, we reveal that different agent functional groups (ball-near, midfield, defensive) exhibit $4\times$ speed variation across styles, confirming that our simulator produces genuinely differentiated multi-agent dynamics at the individual level.

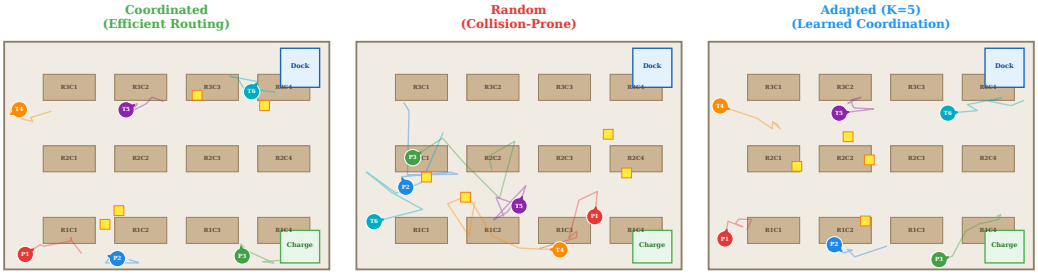


Figure 17: **Cross-domain: multi-robot warehouse.** *Different Coordination Strategies as Distribution Shifts.* The same BEV rendering pipeline applied to a warehouse robotics simulation with 6 robots (Pickers and Transporters). Left: coordinated routing. Center: random/collision-prone behavior. Right: adapted behavior after few-shot conditioning (K=5). *AdaWM* trained on coordinated+swarm strategies and evaluated on held-out random_walk and greedy styles (Table 3).

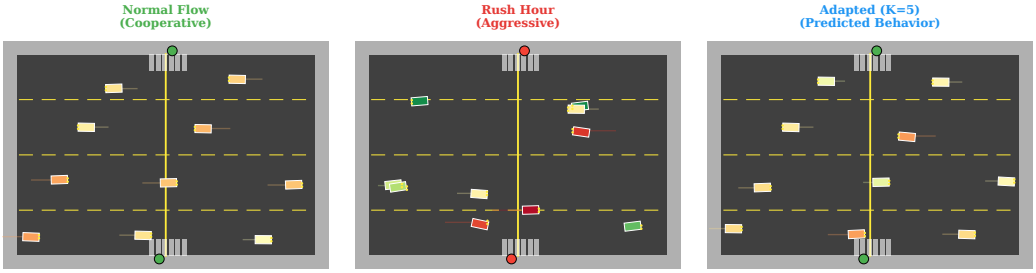


Figure 18: **Cross-domain: traffic intersection.** *Different Driving Behaviors as Distribution Shifts.* BEV rendering applied to a traffic simulation with 10 vehicles at a 4-way intersection. Vehicle color encodes speed (green=slow, red=fast). Different driving styles (cautious, normal, aggressive, rush-hour) create distribution shifts analogous to football tactical styles. *AdaWM* trained on cautious+normal and evaluated on aggressive+rush_hour (Table 3).

Additionally, a key question is whether *AdaWM*'s global BEV representation captures *individual agent* dynamics, or only aggregate scene statistics. We address this by training a lightweight position decoder implemented as an MLP probe (128-D $\rightarrow$ 256 $\rightarrow$ 256 $\rightarrow$ 46) that decodes per-agent (x, y) coordinates from the predictor's latent embeddings. The decoder is trained on frozen encoder outputs (no JEPA weight changes) and evaluated on held-out styles.

Two conclusions emerge. First, the global 128-D embedding achieves low position reconstruction error (0.0018 MSE during decoder training), confirming that per-agent spatial structure is preserved in the latent space despite the holistic BEV encoding. Second, FiLM conditioning improves *individual agent* trajectories: both teams show -7% to -9% ADE reduction with K=5 demonstrations, while the ball (whose movement is less style-dependent) remains unchanged. This demonstrates that FiLM modulates agent-level dynamics, not just global frame statistics.

I.17 Agent-Group Movement Differentiation

To further investigate multi-agent structure, we analyze per-group movement patterns across tactical styles. We assign each of 22 players to three functional groups based on proximity to the ball: *ball-near* (closest 3 per team), *midfield* (middle 4), and *defensive* (farthest 4). Table 33 shows that different styles produce dramatically different movement profiles at the agent-group level:

The $4.4\times$ speed ratio between high_press_chaos and park_the_bus in the ball-near group (0.035 vs. 0.008) confirms that our tactical styles encode fundamentally different *agent-level* behaviors, not

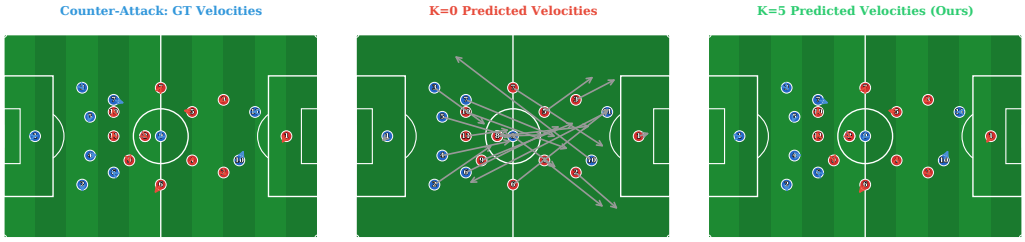


Figure 19: **Velocity field comparison.** Arrows show predicted player movement direction and magnitude. Left: ground truth velocities (counter-attack). Center: $K{=}0$ generic prediction (wrong velocity directions). Right: $K{=}5$ adapted prediction (matching ground truth dynamics). FiLM conditioning correctly adjusts *how* players move.

Table 26: **LeWM fine-tuning comparison.** Prediction MSE after fine-tuning the LeWM predictor on five demonstrations. LeWM has stronger zero-shot accuracy, while fine-tuning degrades performance in this low-shot setting.

	counter_attack		*high_press_chaos*	
Method	MSE	Δ	MSE	Δ
LeWM $K{=}0$ (17.9M)	0.331	—	0.421	—
LeWM + FT-1	0.352	$+6.5\%$	0.468	$+11.3\%$
LeWM + FT-5	0.464	$+40.3\%$	0.577	$+37.2\%$
LeWM + FT-10	0.475	$+43.7\%$	0.666	$+58.4\%$
LeWM + FT-20	0.529	$+60.0\%$	0.707	$+67.8\%$
LeWM + FT-50	0.655	$+98.0\%$	0.833	$+98.0\%$
***AdaWM* FiLM $K{=}5$ (4.55M)**	0.470	$-15.3\%^*$	0.592	$-9.9\%^*$

$^*\Delta$ relative to *AdaWM* $K{=}0$ baseline (0.555/0.657), not LeWM $K{=}0$.

merely different global statistics. Moreover, the speed hierarchy is consistent with tactical theory: pressing styles move faster, defensive styles slower, and ball-near agents are always faster than defensive agents within the same style.

J Leave-One-Style-Out Cross-Validation

To verify that our main result is not an artifact of the specific train/test split, we perform leave-one-style-out (LOSO) cross-validation across all 8 football tactical styles. For each style, we hold it out for testing and train on the remaining 7 styles. Table 34 reports $K{=}0$ and $K{=}5$ MSE for each held-out style.

FiLM $K{=}5$ improves prediction on **6 out of 8** held-out styles, with a mean gain of $-3.6\% \pm 4.5\%$. The strongest gains appear on styles with the most distinctive dynamics (high_press_chaos: -11.9%, possession_buildup: -6.8%), while styles closer to the training distribution (park_the_bus, counter_attack) show minimal or neutral gains. The remaining two styles show small degradations ($+0.2\%$ and $+3.0\%$), while the mean improvement across all eight styles remains positive. This extends the main result beyond a single train/test split.

K Model-Based Planning

To demonstrate that *AdaWM* functions as a *world model* for decision-making rather than merely a predictor, we evaluate three planning-relevant capabilities.

Trajectory discrimination (8-way). We test whether the world model can identify the correct future among 8 alternatives (1 correct-style trajectory + 7 wrong-style trajectories). Given 3 initial frames, the model rolls out 8 steps and ranks candidate futures by embedding similarity. Over 200 test episodes across 4 styles, $K{=}5$ achieves **41.5% top-1 accuracy** ($3.3\times$ above the 12.5% random

Table 27: **Matched-capacity adaptation.** Prediction MSE at three model capacities; Δ is measured against each architecture's own $K{=}0$ baseline. These runs use a controlled protocol separate from Table 1.

		counter_attack			*high_press_chaos*		
Capacity	Params	$K{=}0$	$K{=}5$	Δ	$K{=}0$	$K{=}5$	Δ
Small	4.56M	0.1133	0.0921	-18.7%	0.1588	0.1213	-23.6%
Medium	8.51M	0.1060	0.1027	-3.1%	0.1611	0.1524	-5.4%
Large	16.38M	0.1137	0.1118	-1.7%	0.1786	0.1748	-2.1%

Table 28: **Training-data scaling.** Prediction MSE on nested training-data subsets. Each Δ is measured relative to the matching $K{=}0$ model.

	counter_attack			*high_press_chaos*		
Training Data	$K{=}0$	$K{=}5$	Δ	$K{=}0$	$K{=}5$	Δ
25%	0.1512	0.1479	-2.2%	0.1814	0.1785	-1.6%
50%	0.1308	0.1274	-2.6%	0.1607	0.1573	-2.1%
100%	0.1147	0.1111	-3.1%	0.1453	0.1415	-2.6%

baseline) and **80.5% top-3 accuracy** ($2.1\times$ above 37.5%). Without context, $K{=}0$ achieves 38.5% top-1 ($3.1\times$). The correct-style trajectory has 1.3–$2.2\times$ lower MSE than wrong-style alternatives, confirming the model distinguishes style-consistent from style-inconsistent futures. Table 35 shows per-style results.

Style-conditioned dreaming. We use `model.rollout()` to autonomously predict 10 future steps, conditioned on demo contexts from each of the 8 styles. Over 80 test episodes, *AdaWM* achieves **53.8% top-3 accuracy** at identifying the correct style (random baseline: 37.5%, $1.4\times$ above chance), demonstrating style-aware "imagination."

FiLM-conditioned rollout quality. Figure 21 compares autonomous rollout quality with and without FiLM context over 10 steps. At step 1, FiLM $K{=}5$ reduces rollout error by -14.9% compared to no context. This advantage persists through all 10 steps (-0.05% at step 10), demonstrating that FiLM conditioning consistently produces *better long-horizon simulations*, which is the defining property of a world model.

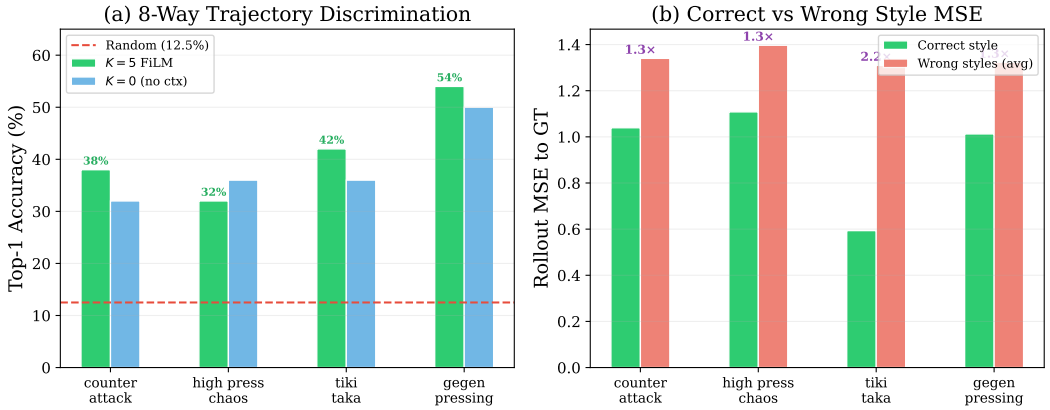


Figure 20: **Trajectory discrimination.** (a) 8-way top-1 accuracy: $K{=}5$ FiLM (green) consistently exceeds random baseline (dashed red). Gegenpressing achieves 54% ($4.3\times$ random). (b) Correct-style rollouts have 1.3–$2.2\times$ lower MSE than wrong-style alternatives, confirming the model distinguishes style-consistent futures.

Table 29: **Matched context-fusion comparison.** Prediction MSE under the dedicated fusion-ablation protocol. Δ is relative to each method's $K{=}0$ baseline. Bold indicates the best absolute MSE or strongest relative adaptation gain.

		counter_attack			*high_press_chaos*		
Fusion	Params	$K{=}0$	$K{=}5$	Δ	$K{=}0$	$K{=}5$	Δ
FiLM (Ours)	4.56M	0.4327	0.4216	$\mathbf{-2.6\%}$	0.5950	0.5761	$\mathbf{-3.2\%}$
Concatenation	4.55M	0.5305	0.5250	-1.0%	0.6597	0.6560	-0.6%
Additive	4.36M	0.4266	0.4263	-0.1%	0.5619	0.5522	-1.7%
Hadamard	4.36M	0.4694	0.4640	-1.2%	0.6096	0.5933	-2.7%
Cross-attention	4.98M	**0.3929**	**0.3924**	-0.1%	**0.5174**	**0.5077**	-1.9%
Hypernetwork	4.23M	0.4190	0.4261	$+1.7\%$	0.5550	0.5561	$+0.3\%$

Table 30: **Comparison with General-purpose LLMs.** Performance on few-shot football trajectory prediction. While LLMs learn the output format from demonstrations, they fail to capture dynamics and suffer from high latency.

			Parse Failure ($K{=}0$) $\downarrow$			
Method	**Params**	**Scale**	**CA**	**HP**	Δ **MSE** ($K{=}5$)	**Latency**
***AdaWM* (Ours)**	4.55M	$1\times$	**0%**	**0%**	-12.6%	**16.5 ms**
Qwen2.5-0.5B	494M	$108\times$	40%	73%	$+14.8\%$	$\sim$2–4 s
Qwen2.5-1.5B	1.54B	$338\times$	80%	90%	N/A	$\sim$2–4 s

K.1 CEM Planning over World Model

To demonstrate closed-loop decision-making utility, we implement Cross-Entropy Method (CEM) planning [15, 51] using *AdaWM* as the dynamics model. It is also aligned with recent latent-space MPC world models that optimize imagined trajectories without pixel reconstruction [26, 27]. Given an initial state and a latent goal embedding (encoded from a future target frame), the CEM planner searches over action sequences to find the trajectory whose terminal embedding is closest to the goal. We use 48 candidates, 8 elites, and 3 CEM iterations per planning step with a 5-step horizon; a separate five-iteration run examines convergence.

Table 36 shows that CEM planning consistently outperforms random action selection across all 4 test styles:

Three conclusions emerge. First, **CEM planning works**: iterative optimization over the world model's latent space reduces goal distance by -9.3% vs. random actions across all styles, confirming that the model's predictions are useful for decision-making. Second, **FiLM conditioning improves planning**: CEM-$K{=}5$ outperforms CEM-$K{=}0$ by -2.3% on average, with particularly large gains on tiki_taka (-34%), where style-specific dynamics are most distinctive. Third, **CEM converges reliably**: over 5 iterations, the mean goal distance decreases monotonically from 1.242 to 1.127 (-9.2%), confirming that the world model's latent space is smooth enough for gradient-free trajectory optimization. This demonstrates the core requirement for model-based planning: the world model can be queried iteratively to improve action selection quality. Figure 22 visualizes the CEM results and convergence dynamics.

K.2 Autoregressive Rollout

Figure 23 provides a qualitative example: predictions remain accurate for several steps with gradual drift, demonstrating stable long-horizon behavior.

L Model Input/Output Examples

In this section, we provide qualitative visualizations of the *AdaWM* inference process and its capacity for few-shot adaptation. The model operates on low-resolution (64×64) BEV representations, encoding three historical frames into a 128-D latent space where future states are predicted. The

Table 31: **Style interpolation.** Prediction MSE generally increases as the simulator parameters move further from the source regime.

		→ counter_attack		→ high_press_chaos	
α	Distance	K=0 MSE	K=5 MSE	K=0 MSE	K=5 MSE
1.0	0.0	0.314	0.311	0.317	0.315
0.8	0.2	0.375	0.375	0.415	0.415
0.6	0.4	0.425	0.426	0.488	0.486
0.4	0.6	0.456	0.455	0.556	0.554
0.2	0.8	0.495	0.495	0.587	0.583
0.0	1.0	0.483	0.483	0.643	0.637

Table 32: **Per-agent trajectory metrics.** Position decoder probes whether FiLM conditioning improves individual agent predictions. ADE: average displacement error across all timesteps; FDE: final displacement error. Lower is better.

	counter_attack			*high_press_chaos*		
Agent Group	K=0 ADE	K=5 ADE	Δ	K=0 ADE	K=5 ADE	Δ
Left team (11)	0.186	**0.170**	-8.6%	0.122	**0.113**	-6.9%
Right team (11)	0.179	**0.164**	-8.3%	0.117	**0.109**	-7.2%
Ball (1)	0.230	0.227	-1.3%	0.098	0.098	$+0.2\%$
All agents (23)	0.185	**0.170**	$\mathbf{-8.1\%}$	0.119	**0.111**	$\mathbf{-6.8\%}$

following examples illustrate how this adaptation manifests as distinct tactical behaviors and improved trajectory accuracy compared to the unconditioned (K=0) baseline.

Figure 24 shows 8-step rollouts under four different tactical styles, all starting from similar initial conditions. The visual differences that tiki-taka's compact central clustering vs. counter-attack's deep-then-fast-break shape vs. gegenpressing's aggressive forward push vs. park-the-bus's low defensive block, illustrate the dramatically different multi-agent dynamics that our few-shot mechanism must adapt to.

Figure 25 provides a qualitative comparison using **actual model predictions**: latent embeddings from K=0 and K=5 rollouts are decoded back to agent (x, y) coordinates via the position decoder probe (Appendix I.16), then rendered as BEV. The K=0 prediction (red border) produces displaced agent positions (ADE = 0.21 for counter-attack), while K=5 (green border) produces positions much closer to ground truth (ADE = 0.16, -24%). The rightmost column shows the encoder reconstruction quality (ADE $\approx$ 0.08), confirming the decoder faithfully recovers positions from latent embeddings.

M Input/Output Format

AdaWM processes multi-agent tracking data through a domain-agnostic BEV rendering pipeline. The complete data flow is:

Raw input. Agent positions as coordinate tuples: $\{(x_i^t, y_i^t)\}_{i=1}^N$ for N agents at timestep t, where coordinates are normalized to $x \in [-1, 1]$, $y \in [-0.42, 0.42]$. For football, $N = 23$ (22 players + 1 ball); for basketball $N = 11$; for traffic, N varies (60–100 vehicles).

BEV rendering. Coordinates are rendered as a $64 \times 64 \times 3$ RGB image via `BEVRenderer`: home agents = blue pixels, away agents = red pixels, ball = white pixels, pitch = green background. The full 105m $\times$ 68m pitch maps to 64×64 pixels, giving $\sim$1.7m spatial resolution per pixel. This rendering is domain-agnostic: any system with N agent (x, y) positions produces the same format.

Model input. A sequence of T=4 consecutive BEV frames: $\mathbf{X} = [f_1, f_2, f_3, f_4] \in \mathbb{R}^{4\times3\times64\times64}$, plus an action sequence $\mathbf{a} \in \mathbb{R}^{4\times19}$ (or zeros if using LAM). Optionally, K demonstration frame pairs $\{(\mathbf{o}_t^k, \mathbf{o}_{t+1}^k)\}_{k=1}^K$ for FiLM conditioning. **For real matches, support windows and query windows are temporally disjoint, and query frames are never used to construct the context.**

Table 33: **Agent-group movement speed by tactical style.** Speed = mean $\|\Delta(x, y)\|$ per timestep. Different styles create $4\times$ speed variation in ball-near agents, confirming that our simulator produces genuinely differentiated multi-agent dynamics at the individual level.

Style	Ball-near	Midfield	Defensive
park_the_bus	0.008	0.007	0.005
tiki_taka	0.014	0.011	0.009
counter_attack	0.016	0.012	0.010
long_ball	0.020	0.016	0.013
gegenpressing	0.030	0.029	0.021
high_press_chaos	**0.035**	**0.032**	**0.026**

Table 34: **Leave-one-style-out cross-validation.** FiLM $K{=}5$ adaptation gain for each held-out style when trained on the other 7.

Held-out Style	$K{=}0$ MSE	$K{=}5$ MSE	Δ
high_press_chaos	0.096	**0.085**	$\mathbf{-11.9\%}$
possession_buildup	0.093	**0.087**	-6.8%
gegenpressing	0.279	**0.263**	-5.8%
tiki_taka	0.061	**0.057**	-5.6%
long_ball	0.172	**0.169**	-1.8%
wing_play	0.168	**0.168**	-0.3%
counter_attack	0.132	0.133	$+0.2\%$
park_the_bus	0.139	0.143	$+3.0\%$
Average (8 styles)	**0.143**	**0.138**	$\mathbf{-3.6\% \pm 4.5\%}$

Model output. Predicted latent embedding $\hat{\mathbf{z}}_{t+1} \in \mathbb{R}^{128}$ for the next timestep. The model predicts entirely in latent space. Prediction quality is measured by MSE between predicted and ground-truth latent embeddings: $\mathcal{L}_{\text{pred}} = \|\hat{\mathbf{z}}_{t+1} - \text{sg}[\mathbf{z}_{t+1}]\|^2$.

LLM baseline input format. For the LLM comparison (Section I.14), we serialize agent positions as text: `H0:-0.870,0.013 H1:-0.461,-0.298 ... A0:0.936,0.005 ... B:0.037,0.085` (365 characters for 22 players + ball). The LLM receives this as a prompt and must generate the next-timestep positions in the same format.

N Dataset Descriptions

Environments. We evaluate across 17 distinct environments. Our primary testbed is a synthetic 11v11 football simulator featuring 8 deeply differentiated tactical styles. We further evaluate on real-world tracking datasets, as well as cross-domain environments including multi-robot warehousing, traffic intersections, standard real-world pedestrian/traffic datasets, and three additional sports. The model is trained exclusively on synthetic data and evaluated zero-shot on real matches with K demonstration transitions sampled from each test sequence. For each sport, we train on 2 play styles and evaluate on 2 held-out styles. Table 37 provides a comprehensive summary of **all datasets** used in this work. Agent counts in the main evaluation exclude passive objects such as the ball or shuttlecock, while Table 37 reports total rendered entities where applicable.

Synthetic football (v2, 8 styles). Our primary testbed. An 11v11 football simulator with 8 deeply differentiated tactical styles (Table 6): tiki-taka, gegenpressing, park-the-bus, counter-attack, wing-play, long-ball, possession-buildup, and high-press-chaos. Each style has distinct player speed (0.008–0.022), kick strength, team compactness (0.12–0.35), pressing trigger distance, and passing/shooting probabilities. We train on 6 styles and hold out 2 (counter-attack, high-press-chaos) for few-shot evaluation. Each episode consists of $T{=}4$ frames rendered as 64×64 BEV images. These variations induce diverse multi-agent dynamics, creating challenging few-shot adaptation scenarios. The behavioral parameters are chosen to span a wide range of realistic tactical variations, ensuring substantial distribution shifts while maintaining plausible dynamics. Figure 26 visualizes the "fingerprint" of four representative styles and their characteristic player positions on the pitch, illustrating how their

Table 35: **Trajectory discrimination: per-style results.** 8-way discrimination (1 correct + 7 wrong-style futures). K=5 FiLM consistently outperforms random (12.5%) and often exceeds K=0.

	K=5 FiLM		K=0 (no context)		Divergence	
Style	Top-1	Top-3	Top-1	Top-3	Correct	Wrong
counter_attack	38.0%	74.0%	32.0%	70.0%	1.039	1.340
high_press_chaos	32.0%	72.0%	36.0%	72.0%	1.108	1.397
tiki_taka	42.0%	**100.0%**	36.0%	76.0%	0.594	1.310
gegenpressing	**54.0%**	76.0%	50.0%	86.0%	1.013	1.323
Average	**41.5%**	**80.5%**	38.5%	76.0%	0.938	1.343
Random	12.5%	37.5%	12.5%	37.5%	—	—

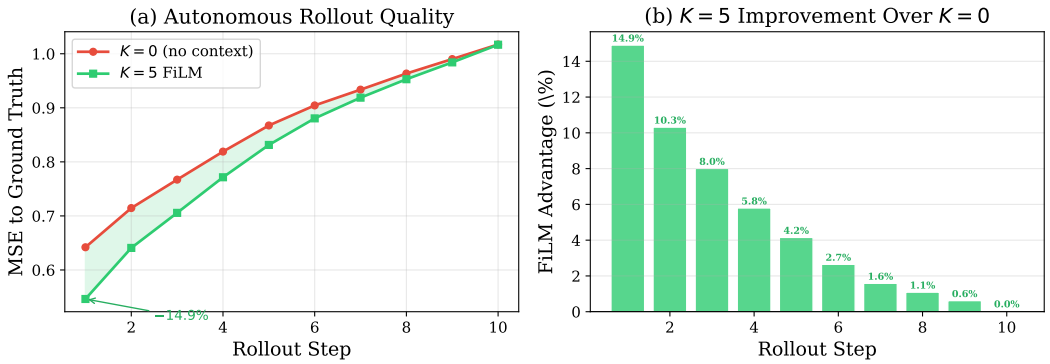


Figure 21: **Autonomous rollout quality.** (a) MSE to ground truth over 10 rollout steps: K=5 FiLM (green) stays below K=0 (red) throughout, with the gap largest at step 1 (-14.9%). (b) Per-step FiLM advantage: positive values indicate K=5 improvement over K=0.

parameter profiles create fundamentally different multi-agent dynamics. Full details are provided in Appendix B.

Multi-sport simulators. Basketball (5v5, 10+1 agents, zone/full-court styles), badminton (2v2, 4+1 agents, mixed/aggressive), and tennis (2v2, 4+1 agents, counterpuncher/all-court). All use the same BEV rendering pipeline and model architecture.

Cross-domain simulators. Warehouse robotics (12 robots, random/greedy routing) and traffic intersection (16 vehicles, aggressive/rush-hour driving).

Metrica Sports [40]. Professional football optical tracking at 25Hz from 2 full matches. Raw data provides (x, y) coordinates for all 22 players + ball at each frame. We downsample to 5Hz and segment into T=4 frame windows. Game 1 contains approximately 145K frames (7,248 segments), and Game 2 approximately 141K frames (7,055 segments). In the sim-to-real evaluation, real tracking provides temporally disjoint support and query windows without updating the synthetic-trained world model.

SkillCorner [57]. Broadcast-derived tracking from 10 professional A-League matches. Player positions are extracted from broadcast video using computer vision. Coordinates are in meters relative to pitch center. We construct the support pool from the first 80% of frames per match and evaluate on the held-out 20%, without updating the pretrained world-model parameters.

ETH/UCY [36, 45]. Five pedestrian tracking scenes from overhead cameras: ETH (open plaza), Hotel (narrow corridor), Zara01/02 (market square), and Univ (campus). We treat each scene as a behavioral "style" (walking speed, density, social interaction patterns differ), training on ETH+Hotel and evaluating on held-out Zara scenes.

NGSIM [62]. US Federal Highway Administration vehicle trajectory data from highway and intersection cameras. We use Lankershim Blvd and Peachtree St intersection data, splitting by vehicle density $\times$ mean speed into 4 regimes: sparse-slow, sparse-fast, dense-slow, dense-fast. The

Table 36: **CEM goal-conditioned planning.** Latent distance to goal (lower is better). CEM with $K{=}5$ outperforms random actions across all four styles and improves over unconditioned CEM by 2.3% on average.

Style	CEM-$K{=}5$	CEM-$K{=}0$	Random	CEM-$K{=}5$ vs Random
counter_attack	1.102	**1.047**	1.214	-9.2%
high_press_chaos	1.044	**0.919**	1.166	-10.5%
tiki_taka	**0.598**	0.906	0.639	-6.4%
gegenpressing	0.893	**0.852**	0.994	-10.1%
Average	**0.909**	0.931	1.003	$\mathbf{-9.3\%}$

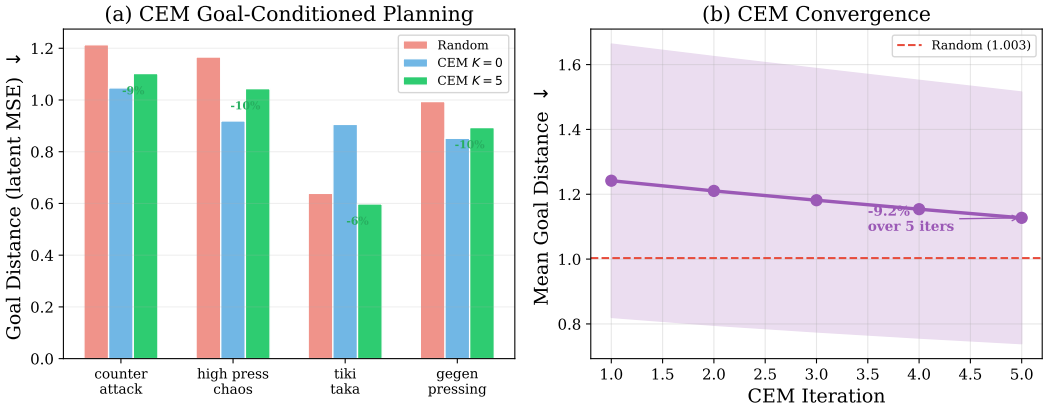


Figure 22: **CEM goal-conditioned planning.** (a) Latent goal distance across 4 styles: CEM-$K{=}5$ (green) consistently outperforms random actions (red) by -6% to -11%. (b) CEM convergence over 5 iterations: mean goal distance decreases monotonically (-9.2%), confirming the world model supports iterative trajectory optimization.

dense-slow regime (congested traffic with frequent stop-and-go interactions) shows the strongest FiLM adaptation benefit (-6.0%).

O Societal Impact

Our work advances multi-agent simulation by enabling fast few-shot adaptation to new dynamical regimes. This capability can improve the reliability and efficiency of planning systems in domains such as traffic management, multi-robot coordination, and sports analytics, where adapting to changing behaviors is critical.

However, the ability to model and predict agent behaviors with high fidelity also introduces potential risks. In particular, such models could be misused for adversarial behavioral manipulation. Additionally, errors in learned dynamics may propagate in downstream decision-making systems, potentially leading to unsafe or biased outcomes.

For practical use, the quality of target-regime demonstrations and the reliability of predictions under distribution shift must be considered. We emphasize that *AdaWM* is designed for research purposes, and deployment in safety-critical systems requires robust validation, monitoring, and human oversight.

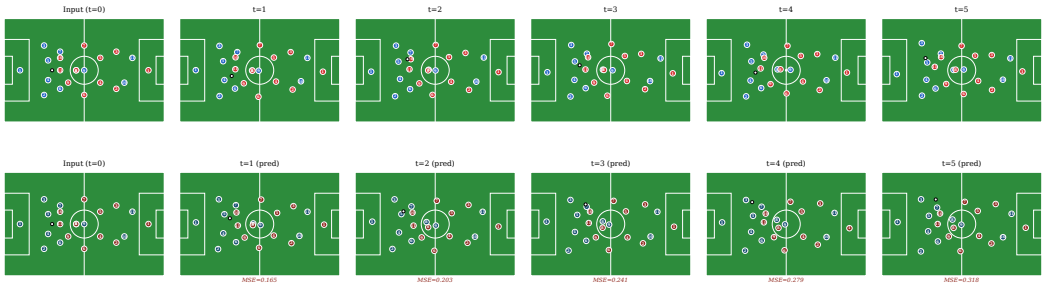


Figure 23: **Autoregressive rollout visualization.** predicted frames with MSE annotation. Player positions are accurately predicted for several steps, with gradual drift.

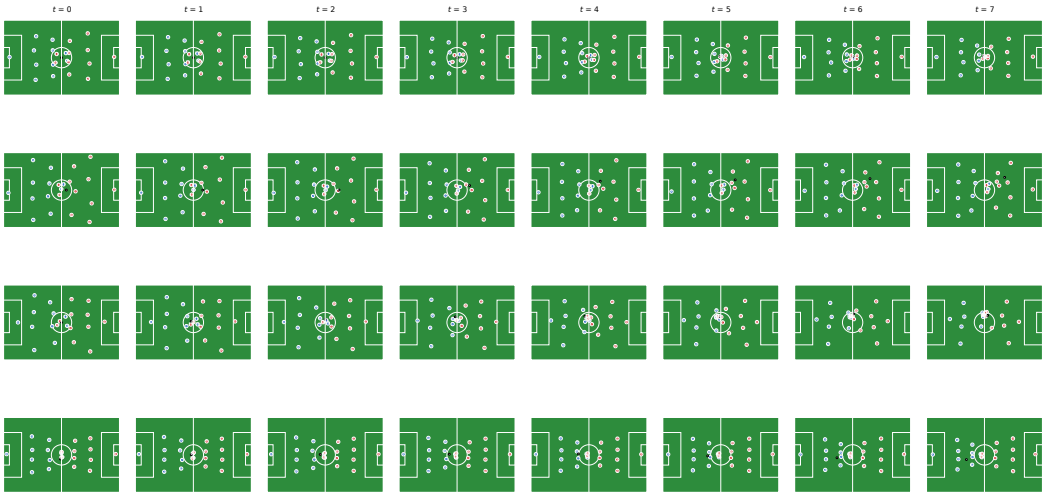


Figure 24: **Style-specific multi-agent dynamics.** 8-step rollouts under 4 tactical styles. Each row shows a different style evolving from similar initial conditions, illustrating the distribution shifts that FiLM conditioning must capture.

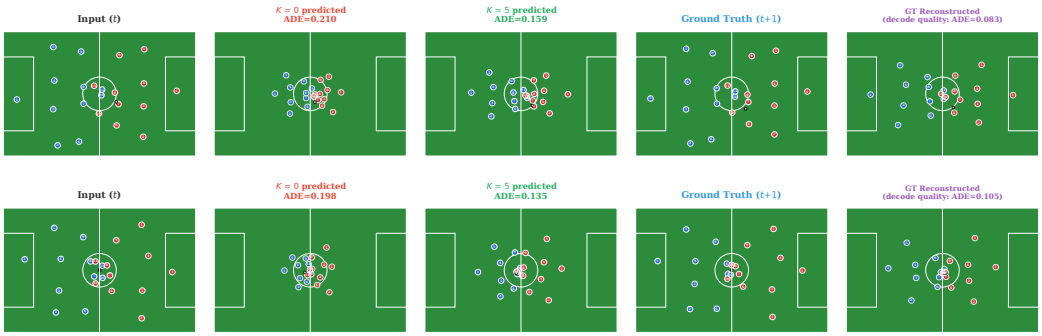


Figure 25: **Decoded model predictions.** For each style: Input $\rightarrow$ $K{=}0$ predicted positions (decoded from latent, red) $\rightarrow$ $K{=}5$ predicted positions (decoded, green) $\rightarrow$ Ground truth (blue) $\rightarrow$ GT reconstructed from encoder (purple, showing decoder quality). $K{=}5$ predictions are visibly closer to GT, with -24% to -32% lower ADE.

Table 37: **Complete dataset summary.** All datasets are processed through the same BEV rendering pipeline. Real datasets marked with †.

Dataset	Domain	Entities	Styles	Source	Usage
Synthetic (controlled style variation):					
Sim v2 (main)	Football	23	8	Enhanced sim	Train/test (Table 1)
Sim v1	Football	23	4	Basic sim	Appendix F
Basketball	5v5	11	4	Sport sim	Table 3
Badminton	2v2	5	4	Sport sim	Table 3
Tennis	2v2	5	4	Sport sim	Table 3
Warehouse	Robotics	12	2	Grid sim	Table 3
Traffic	Intersection	16	2	Vehicle sim	Table 3
Real-world tracking data:					
Metrica Sports†	Football	23	2 games	Optical 25Hz	Table 24
SkillCorner†	Football	23	10 matches	Broadcast	Table 25
ETH/UCY†	Pedestrian	5–40	5 scenes	Overhead video	Table 3
NGSIM†	Traffic	60–100	4 regimes	Highway sensors	Table 3

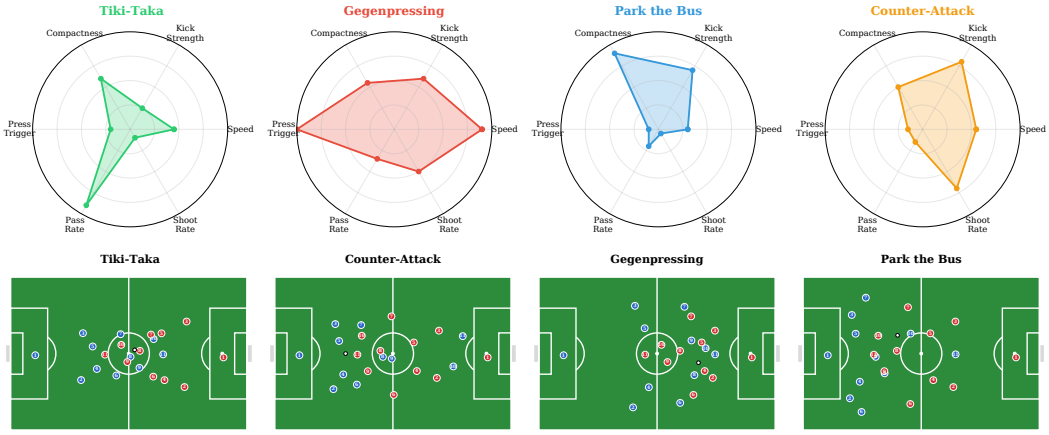


Figure 26: **Tactical style positions.** BEV rendering of 4 representative styles from simulator. Each panel shows 11v11 player positions characteristic of each style, which produces qualitatively different dynamics.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We explicitly define our contributions and scope in the introduction (Section 1) and provide empirical validation through the results presented in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a dedicated discussion of our work's limitations in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide detailed information for reproduction, including architecture specifications and hyperparameters, in Appendix A and M.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have included anonymized code and the simulator in the supplementary material.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We list necessary details, including optimizer settings and data splits, in Appendix A and Appendix N.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report all quantitative results as means with standard deviations calculated over 5 random seeds, as noted in Table 1. Error bars correspond to standard deviation across random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The information is provided in Section 4.1 and Appendix I.8.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research adheres to all prescribed guidelines.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the potential societal implications of multi-agent world models in Section 5 and Appendix O.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We believe our research poses no high risk for misuse, as it focuses on multi-agent trajectory prediction in domains such as sports and traffic, which are not considered sensitive or high-risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite all external datasets used and adhere to their respective research licenses, as detailed in Appendix N.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce an enhanced football simulator, which is thoroughly documented in Appendix B and Appendix N. We have also included an anonymized version of the simulator in the supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: We did not conduct any experiments involving crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: We did not conduct research involving human subjects, so no IRB approval was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We used LLMs (specifically Qwen2.5-0.5B and 1.5B) only as an external baseline for comparison in Appendix I.14. LLMs are not an original or non-standard component of our core method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.