SPIKING ACTOR-CRITIC WITH TEMPORAL DENDRITIC HETEROGENEITY FOR ASYNCHRONOUS MULTI-AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Asynchronous multi-agent reinforcement learning requires agents to coordinate while making decisions at different macro-action completion times. Information exchange can supplement incomplete local observations, and yet under asynchronous execution, a receiver's current observation can be combined with teammate messages generated at the same or earlier times. This temporal mismatch can make it difficult to retain useful teammate information without allowing outdated information to influence current decisions. To integrate this asynchronous information, we propose CoSpAC, a compartmentalized spiking actor-critic framework inspired by temporal dendritic heterogeneity. Each actor processes local observations and aggregated teammate messages through separate dendritic compartments with distinct decay factors. In this way, CoSpAC provides different temporal integration timescales for observations and messages from different decision times, which are subsequently combined through somatic dynamics for action selection. Additionally, CoSpAC exchanges teammate information at agents' individual decision events and optimizes message generation through a receiver-oriented objective based on its influence on receiving agents' policies. Experiments across asynchronous multi-agent tasks show that CoSpAC outperforms state-of-the-art asynchronous MARL approaches, demonstrating the effectiveness of the proposed framework for asynchronous cooperation.

1 INTRODUCTION

Multi-Agent Reinforcement Learning (MARL) provides a general framework for learning cooperative policies among multiple interacting agents (Yu et al., 2022; Dou et al., 2026; Rashid et al., 2020). In many real-world multi-agent systems, however, synchronized decision making cannot be assumed (Zhang et al., 2023). For example, robots in cooperative navigation (Şenbaşlar & Sukhatme, 2022), object transportation (Yuan & Mahmood, 2022), or manipulation (Wang et al., 2025) may execute high-level behaviors with substantially different and state-dependent durations. In this scenario, an agent that has completed its current behavior should be able to make its next decision without waiting for all teammates to reach the same decision point (Zhang & Zhang, 2025; Hoang et al., 2018).

Temporally extended actions, or macro-actions, provide a natural abstraction for these behaviors, but also cause agents to terminate actions and request new decisions at different times (Amato et al., 2019). This gives rise to asynchronous multi-agent reinforcement learning, where agents make decisions according to their own action-completion times while still needing to coordinate toward a shared objective (Jung et al., 2025; Xiao et al., 2020b;a). However, in realistic scenarios, agents act on incomplete local information, a limitation compounded by asynchrony that places decision-relevant information from different agents in distinct temporal contexts (Dolan et al., 2025; Yu et al., 2023). One effective way to support cooperation is to exchange information among agents, allowing them to supplement their local experience with information from teammates (Sheng et al., 2022). Under asynchronous execution, however, agents generate and access messages according to their own decision schedules. At a receiver's decision event, a newly available local observation may therefore be combined with teammate messages generated at earlier times. Observations and messages may differ in how long they remain useful for coordination. Applying a common temporal

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

integration rule can discard useful teammate context too quickly or retain outdated information for too long, thereby impairing coordinated decisions. This motivates separate temporal integration of local and communicated information before they are combined for action selection.

Biological neurons integrate synaptic inputs through membrane dynamics and communicate through spikes. Spiking neural networks (SNNs) capture these processes to retain and integrate information over time (Brette & Gerstner, 2005). At the network level, brain regions exchange information while exhibiting different temporal dynamics, supporting distributed processing across multiple timescales (Chaudhuri et al., 2015). These interactions are shaped by how individual neurons integrate inputs from different pathways, with signals from other cortical areas undergoing local processing in distinct dendritic compartments before influencing neuronal output. Dendritic computation thus provides a cellular mechanism for integrating information exchanged between brain regions (Fişek et al., 2023). Inspired by dendritic organization and temporal heterogeneity, multi-compartment SNNs maintain separate membrane states with distinct decay dynamics and have been explored for learning multi-timescale dynamics (Zheng et al., 2024). However, how such compartmentalized temporal processing can support asynchronous multi-agent cooperation by integrating local observations and teammate messages remains largely unexplored.

Motivated by the distributed and compartmentalized organization of biological information processing, we propose a compartmentalized spiking actor-critic (CoSpAC) framework with two dendritic compartments for asynchronous MARL. Each dendritic compartment has its own learnable decay factor, allowing the actor to learn separate temporal integration timescales for local observations and teammate messages. The states are consequently combined at the soma for action selection. This design allows the actor to retain useful teammate information while updating its local memory with new observations. To support asynchronous cooperation, communication is associated with agents' decision events, and message generation is optimized according to its impact on receiving agents through a receiver-oriented objective.

Our contributions can be summarized as follows,

- We introduce a neuroscience-inspired view of asynchronous multi-agent cooperation as compartmentalized temporal information processing. Under this view, local observations and teammate information are integrated through distinct temporal dynamics to support decisions based on asynchronously generated information.
- We develop a multi-compartment spiking actor-critic model that balances local and communicated information through separate dendritic states with distinct decay factors, and learns event-based communication through a receiver-oriented objective.
- Extensive experiments on asynchronous multi-agent tasks show that our method outperforms state-of-the-art asynchronous MARL approaches across various scenarios.

2 Related Works

2.1 Asynchronous Multi-Agent Reinforcement Learning

Early studies on asynchronous multi-agent decision making can be traced to the Macro-Action Decentralized Partially Observable Markov Decision Process (MacDec-POMDP), which extends the Dec-POMDP framework with temporally extended macro-actions (Amato et al., 2015). This formulation was subsequently adopted in multi-robot planning and coordination problems (Amato et al., 2015). A major step toward modern asynchronous MARL was made by Xiao et al. (2022), who extended actor-critic learning to asynchronously terminating macro-actions under different training paradigms. Subsequent work applied asynchronous MARL to real-time multi-robot exploration and heterogeneous robot cooperation, and further studied deep asynchronous reinforcement learning under partial observability (Yu et al., 2023; Zhang et al., 2023; Xiao et al., 2025).

More recent studies have focused on learning issues that arise specifically from asynchronous trajectories. Jung et al. (2025) showed that directly aligning agent-specific trajectories can introduce temporal misalignment during centralized training, and proposed agent-centric processing for asynchronous actor-critic learning. Following this direction, ToMacVF studied temporal credit assignment and value factorization over macro-action execution (Zhang & Zhang, 2025), while AsynCoMARL investigated cooperative learning with infrequent and asynchronous communication (Dolan

et al., 2025). Our work focuses on temporal processing within decentralized actors, using distinct dendritic dynamics to integrate local observations and teammate information generated on different decision schedules.

2.2 Spiking Neural Networks for Reinforcement Learning

Spiking neural networks have been increasingly integrated into RL to exploit stateful neural dynamics and potential energy efficiency, covering value-based learning (Yuan et al., 2019), policy-gradient methods (Yang et al., 2024), and multi-agent cooperation (Li et al., 2023). Recent studies have addressed several challenges specific to spiking dynamics, including the mismatch between internal neural simulation and environment decision steps, surrogate-gradient optimization (Van den Berghe et al., 2026), and unstable normalization statistics in online reinforcement learning (Xu et al., 2026b;a). These studies demonstrate the potential of SNNs for sequential and cooperative decision making, while the use of compartmentalized spiking dynamics to represent and integrate individual and collaborative information under asynchronous decision schedules remains underexplored.

2.3 Multi-Compartment Spiking Neural Networks

Conventional spiking neuron models typically abstract a biological neuron as a single point-like computational unit, while biological neurons exhibit rich dendritic structures in which different compartments can receive distinct inputs, maintain heterogeneous temporal dynamics, and interact before shaping somatic responses. Motivated by these neuroscientific observations, multi-compartment SNNs have introduced soma-dendrite separation (Zhang et al., 2024; 2025), dendritic temporal heterogeneity (Zheng et al., 2024), and nonlinear cross-compartment integration to improve long-term dependency modeling (Feng et al., 2025) and multi-timescale information processing (Ma et al., 2026).

Multi-compartment computation has also begun to appear in reinforcement learning. Existing studies use dendritic compartments to integrate different decision-related inputs in distributional RL, or to enhance long-term memory and predictive modeling in spiking world models (Sun et al., 2025b;a; Liang et al., 2025). These works mainly exploit dendritic computation to enrich the temporal and representational capacity of individual spiking agents. Building on these advances, we use heterogeneous dendritic dynamics to integrate local observations and available teammate messages generated on different decision schedules. The two input pathways undergo separate temporal integration before their states are combined for coordination.

3 Preliminaries

Reinforcement Learning and Asynchronous MARL. We consider a fully cooperative multi-agent reinforcement learning problem with N agents, indexed by $\mathcal{N} = \{1, \dots, N\}$. Following the Macro-Action Decentralized Partially Observable Markov Decision Process (MacDec-POMDP) formulation (Xiao et al., 2022; Jung et al., 2025), the environment is characterized by a global state $s_t \in \mathcal{S}$, local observations $o_t^i \in \mathcal{O}_i$, a shared reward r_t, and a discount factor $\gamma \in [0, 1)$. Each agent selects temporally extended macro-actions from $\mathcal{M}_i$. The joint objective is to maximize the expected discounted return as follows,

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{\pi}} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right], \qquad \boldsymbol{\pi} = \{\pi_{\theta_i}\}_{i=1}^{N}. \tag{1}$$

Unlike synchronous MARL, an agent selects a new macro-action only when its current macro-action terminates. We denote by τ_k^i the environment time at which agent i makes its k-th high-level decision, and by $d_k^i = \tau_{k+1}^i - \tau_k^i$ the duration of the corresponding macro-action. Since d_k^i can vary across agents and decisions, agents generally have different decision schedules. At environment time t, the set of agents that require a new decision is $\mathcal{D}(t) = \left\{ i \in \mathcal{N} \mid \exists k,\ \tau_k^i = t \right\}$, while agents outside $\mathcal{D}(t)$ continue executing their current macro-actions. Let x_k^i denote the observation feature available to agent i at τ_k^i, $a_k^i \in \mathcal{M}_i$ its selected macro-action, and r_k^i the reward associated with this macro-action transition. The agent-centric asynchronous trajectory is written as follows,

$$H^i = \left\{ \left(x_k^i, a_k^i, r_k^i, \tau_k^i \right) \right\}_{k=1}^{K_i}, \tag{2}$$

where K_i may differ across agents and the same token index k does not imply the same environment time. During decentralized execution, agent i selects $a_k^i \sim \pi_{\theta_i}(\cdot \mid H_{<k}^i, x_k^i)$. Under centralized training with decentralized execution, a centralized critic may additionally use the joint asynchronous histories while evaluating the decision of a particular agent. We write the corresponding agent-centric value estimate as $V_{\phi_i,k} = V_{\phi_i}\left(H^{1:N}; \tau_k^i\right)$, and denote the associated advantage estimate by A_k^i. Following PPO-based asynchronous actor-critic methods (Schulman et al., 2017; Jung et al., 2025), the actor is optimized with the clipped surrogate objective

$$\mathcal{L}_{\mathrm{PPO}}^i = -\mathbb{E}_k\left[\min\left(\rho_k^i A_k^i, \mathrm{clip}\left(\rho_k^i, 1-\epsilon, 1+\epsilon\right) A_k^i\right)\right], \tag{3}$$

where

$$\rho_k^i = \frac{\pi_{\theta_i}(a_k^i \mid H_{<k}^i, x_k^i)}{\pi_{\theta_i^{\mathrm{old}}}(a_k^i \mid H_{<k}^i, x_k^i)}. \tag{4}$$

Spiking and Multi-Compartment Neural Dynamics. Spiking neural networks (SNNs) process temporal information through the membrane dynamics of spiking neurons. The dynamics of a Leaky Integrate-and-Fire (LIF) neuron can be written as follows,

$$\tilde{h}_q = \lambda h_{q-1} + I_q, \qquad s_q = \Theta(\tilde{h}_q - \vartheta), \qquad h_q = \tilde{h}_q - \vartheta s_q, \tag{5}$$

where h_q and $\tilde{h}_q$ denote the post-reset and pre-reset membrane potentials, λ is the membrane decay factor, ϑ is the firing threshold, and s_q is the emitted spike. We use q to index the internal spiking steps within one macro-action decision.

Conventional LIF neurons maintain a single membrane state, whereas multi-compartment spiking neurons maintain multiple functionally specialized membrane states for different input pathways (Zhang et al., 2024; Feng et al., 2025; Sun et al., 2025a). For a neuron with C compartments, membrane states can be defined as follows,

$$\mathbf{h}_q = \left[h_q^{(1)}, \ldots, h_q^{(C)}\right]. \tag{6}$$

The compartment states evolve separately and are subsequently integrated for neuronal output.

4 METHOD

4.1 OVERVIEW OF THE PROPOSED FRAMEWORK

Inspired by communication between brain regions and dendritic integration, CoSpAC combines information exchange between spiking actors with compartmentalized temporal processing within each actor. As illustrated in Figure 1, each actor maintains separate dendritic states for local observations and available teammate messages, with distinct learnable decay factors governing their temporal integration. The two states are then combined through somatic dynamics for action selection.

4.2 MULTI-COMPARTMENT SPIKING ACTOR

4.2.1 FUNCTIONAL COMPARTMENTALIZATION

In asynchronous MARL, each agent needs to make decisions based on both its own interaction history and information received from teammates. However, these two types of information originate from different temporal processes. The local observations are generated along the agent's own execution trajectory, while teammate information is produced according to other agents' decision schedules. Local observations are updated at the receiving agent's decision events, whereas available teammate messages may originate from earlier decisions of other agents. We therefore assign these two input streams to separate dendritic pathways, allowing their temporal responses to be regulated through distinct decay factors. This organization allows each actor to maintain its local temporal state while integrating information from other agents through a separate dendritic pathway. The two dendritic states are transmitted to a shared soma, which integrates information and generates spikes.

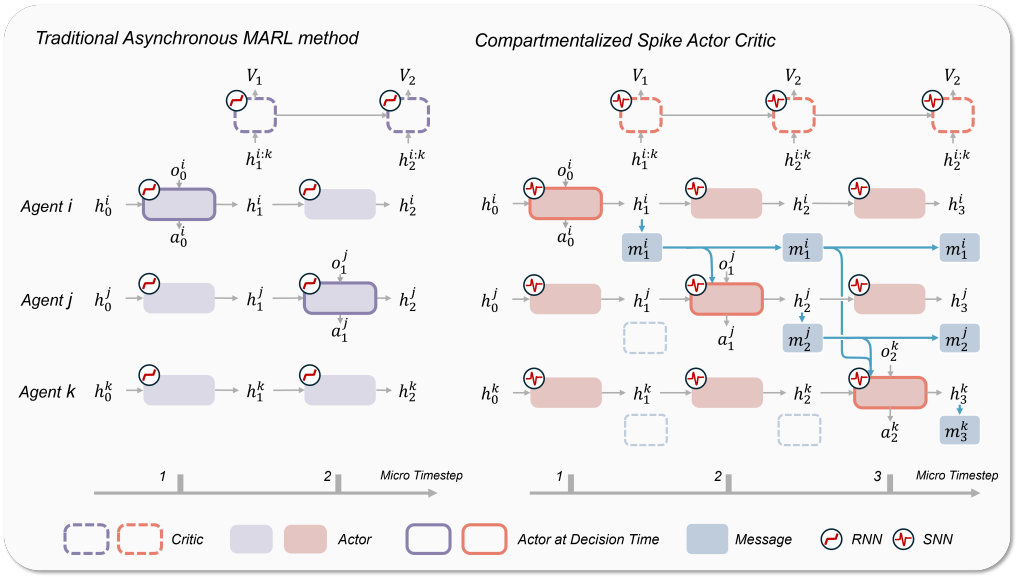

Figure 1: Overview of CoSpAC compared with asynchronous MARL using local recurrent actors. CoSpAC incorporates teammate messages generated at different decision times and processes local and communicated information through distinct dendritic dynamics before somatic integration for action selection.

4.2.2 Dendritic Compartment Dynamics

For agent i at its k-th decision step, we denote the internal spiking step within this decision by q. The proposed actor contains two dendritic compartments, including a local compartment and a communication compartment. The local compartment receives the agent's current observation feature x_k^i and maintains the temporal dynamics of the agent's own experience. Its membrane potential evolves as follows,

$$h_{k,q}^{i,l} = \lambda_l h_{k,q-1}^{i,l} + W_l x_k^i, \tag{7}$$

where $h_{k,q}^{i,l}$ denotes the membrane state of the local compartment, λ_l is the membrane decay factor, and W_l projects observations into the compartment space, superscript l represents *local*. The residual membrane potential at decision step $k-1$ is preserved and propagated to the next decision step k as $h_{k,0}^{i,*} = h_{k-1,Q}^{i,*}$. Meanwhile, the communication compartment receives aggregated teammate information $\tilde{m}_k^i$ and maintains an independent temporal state as follows,

$$h_{k,q}^{i,c} = \lambda_c h_{k,q-1}^{i,c} + W_c \tilde{m}_k^i, \tag{8}$$

where $h_{k,q}^{i,c}$ represents the membrane state of the communication compartment and λ_c controls its temporal integration, superscript c represents *communication*.

Both decay factors, λ_l and λ_c, are learnable and are optimized jointly with the other actor parameters using the policy loss. They control how long the local and communication compartments retain information, respectively. We parameterize the decay factors as $\lambda_l = \sigma(\alpha_l)$ and $\lambda_c = \sigma(\alpha_c)$, where α_l and α_c are trainable parameters and σ is the sigmoid function. The resulting compartment states are then transmitted to the soma compartment as dendritic inputs.

4.2.3 Somatic Integration

The soma compartment integrates membrane potentials from different dendritic compartments and generates output spikes for action selection. Specifically, the somatic input current is computed as follows,

$$I_{k,q}^{i,s} = W_s[h_{k,q}^{i,l}; h_{k,q}^{i,c}], \tag{9}$$

where $[\cdot;\cdot]$ denotes feature concatenation, superscript s represents *soma*. The somatic membrane dynamics follow the LIF formulation as follows,

$$\tilde{h}^{i,s}_{k,q} = \lambda_s h^{i,s}_{k,q-1} + I^{i,s}_{k,q}, \qquad s^{i,s}_{k,q} = \Theta(\tilde{h}^{i,s}_{k,q} - \vartheta_s), \qquad h^{i,s}_{k,q} = \tilde{h}^{i,s}_{k,q} - \vartheta_s s^{i,s}_{k,q}. \tag{10}$$

The final somatic membrane representation is used to generate the action distribution. Somatic integration combines the two dendritic states after their respective temporal processing, allowing local observations and available teammate information to jointly inform the current action.

4.3 Asynchronous Communication Integration

Event-Time Message Generation In asynchronous MARL, agents operate under different decision times. Therefore, communication should be associated with individual decision events rather than synchronized environment steps. When agent i makes its k-th decision at time τ^i_k, it generates a message as follows,

$$m^i_k = g_\psi(h^{i,l}_{k,Q}, \tau^i_k), \tag{11}$$

where $h^{i,l}_{k,Q}$ is the sender's local dendritic state after Q internal steps. Each message is generated from this state at the sender's decision event and encodes its local observation history.

Temporal Message Retrieval and Aggregation At decision time τ^i_k, agent i retrieves the latest available message from each teammate. Let $\hat{m}^j(t)$ denote teammate j's latest message available at time t, or a zero vector if none is available. The message is generated as follows,

$$\mathcal{M}_i(\tau^i_k) = \{\hat{m}^j(\tau^i_k) \mid j \neq i\}, \qquad \tilde{m}^i_k = Agg(\mathcal{M}_i(\tau^i_k)). \tag{12}$$

Receiver-Oriented Communication Learning The effectiveness of a message should be determined by its influence on the receiving agent's decision rather than by the sender alone. Therefore, we optimize the message generation process according to the receiver policy objective. Specifically, the message-related objective is formulated as follows,

$$\mathcal{L}^\psi_{\text{msg}} = -\mathbb{E}[A^i_k \log \pi_i(a^i_k | H^i, \tilde{m}^i_k)], \tag{13}$$

where A^i_k denotes the advantage of the receiving agent. This objective encourages messages that directly improve cooperative decision making.

Practical Algorithm To preserve temporal information during centralized value estimation over asynchronous macro-action trajectories, we further employ an SNN-based recurrent critic, which maintains a membrane state and estimates the value function as follows,

$$V_{\phi,k} = V_\phi\left(H^{1:N}; \tau^i_k\right). \tag{14}$$

Additionally, the decay parameters are optimized jointly with the other actor parameters through the policy objective.

5 Experiments

5.1 Experimental Setup

Benchmarks We evaluate CoSpAC on **BoxPushing** (Xiao et al., 2022), **Overcooked** (Wu et al., 2020), and **Overcooked-Rand** (Wu et al., 2020). These environments require agents to coordinate under partial observability, with macro-actions terminating at agent-specific decision times. BoxPushing involves two agents to cooperate to push boxes into a goal region on 6×6, 8×8, and 10×10 grids, where moving the higher-reward large box requires coordinated actions. Overcooked involves three agents with local observations cooperating to prepare and deliver salads across three 7×7 kitchen layouts, while Overcooked-Rand randomizes agent and object positions at the beginning of each episode. We run each experiment with different random seeds. Curves show the mean, with shaded regions indicating one standard deviation.

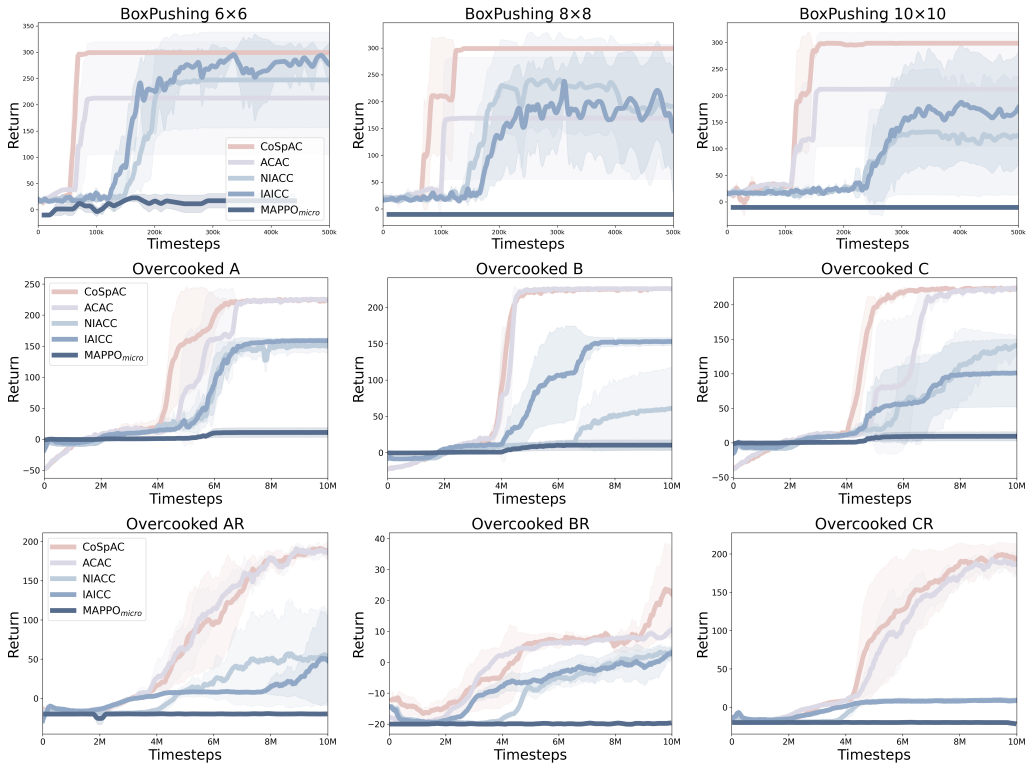

Figure 2: Learning curves on BoxPushing (top), Overcooked A-C (middle), and Overcooked-Rand A-C (bottom). Compared with ANN-based baselines, the SNN-based CoSpAC achieves higher sample efficiency and final returns on BoxPushing, and matches or exceeds ACAC's returns on the Overcooked benchmarks.

Baselines We compare CoSpAC with **ACAC** (Jung et al., 2025), **Mac-NIACC** (Xiao et al., 2022), **Mac-IAICC** (Xiao et al., 2020a), and **MAPPO** (Yu et al., 2022). CoSpAC uses SNN as policy networks, whereas all baselines use conventional artificial neural networks (ANNs). **Mac-NIACC** trains decentralized actors using a single centralized critic, whereas **Mac-IAICC** assigns an individual centralized critic to each actor. **ACAC** encodes each agent's asynchronous history separately and aggregates the resulting representations for centralized value estimation. **MAPPO** applies multi-agent PPO with centralized value estimation and decentralized policies. In this work, CoSpAC, ACAC, Mac-NIACC, and Mac-IAICC operate on macro-actions, while MAPPO operates on micro-actions. We denote this baseline as $\mathrm{MAPPO}_{\mathrm{micro}}$.

5.2 Main Results

Figure 2 shows the main results on Overcooked and BoxPushing. It is clear that macro-action methods generally achieve higher returns than the micro-action method $\mathrm{MAPPO}_{\mathrm{micro}}$, consistent with temporal abstraction facilitating the discovery of coordinated behaviors. CoSpAC further improves sample efficiency and final returns over the macro-action baselines, with a larger performance gap on the larger grids. On Overcooked and Overcooked-Rand, CoSpAC matches or exceeds the returns of ACAC. These results show that the proposed SNN framework, which integrates local observations and teammate messages through distinct dendritic dynamics, is competitive with ANN-based approaches for asynchronous cooperation.

5.3 Performance under Increased Asynchrony

We compare CoSpAC and ACAC on Overcooked under the standard macro-action setting ***Macro*** and two additional conditions. Figure 3 illustrates three action-timing settings. In synchronous MARL (left), agents select actions at common decision steps. In *Macro* (middle), agents execute

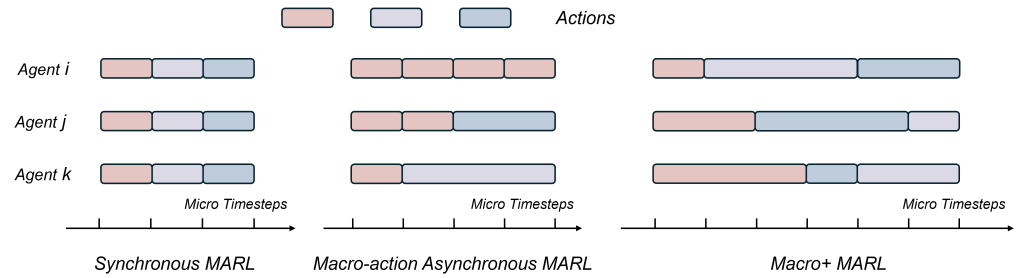

Figure 3: Different asynchrony setting in synchronous MARL (left), Macro (middle), and Macro+ (right). Macro+ introduces agent-specific execution durations, allowing identical macro-actions to terminate at different times.

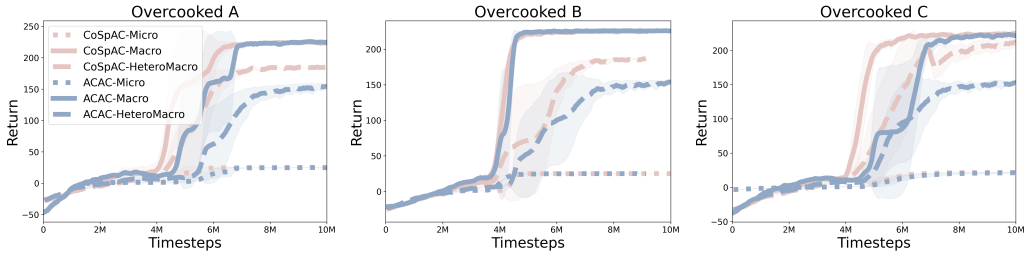

Figure 4: Performance on Overcooked under different action-timing settings. Maximum returns are comparable with standard macro-actions, whereas CoSpAC achieves higher maximum returns under heterogeneous durations in all three tasks.

temporally extended actions and make new decisions upon completion, producing asynchronous decision times as action durations vary. In *Macro+* (right), we additionally introduce agent-specific duration multipliers, so that even the same macro-action with the same nominal duration can finish at different times across agents. This creates further differences in action progress and message update times. Policies must therefore coordinate with teammates whose actions are still ongoing while integrating current local observations with messages generated at different decision events, placing greater demands on asynchronous information processing for cooperation. In the ***Macro+*** setting, we scale the nominal execution durations of the three agents by 0.6, 0.8, and 1, respectively.

Figure 4 shows that CoSpAC and ACAC achieve comparable returns with micro-actions across three tasks. With standard macro-actions, the two methods reach similar maximum returns, while CoSpAC has higher sample efficiency on tasks A and C and comparable sample efficiency on layout B. Under different execution durations, CoSpAC achieves higher maximum returns on all three tasks, and its sample efficiency remains higher on A and C and comparable on B. Although different durations reduce returns for both methods, CoSpAC's advantage extends to maximum returns across all tasks. These results indicate that CoSpAC better handles asynchronous macro-action coordination when agents differ in execution time, benefiting from its separate temporal processing of local observations and teammate information. Additional results with randomly sampled execution-duration multipliers are reported in Appendix A.3.

5.4 Effects of Heterogeneous Compartments

We compare CoSpAC with two variants under standard and heterogeneous macro-action durations. The single-compartment variant ***SC*** processes only local observations, while the homogeneous two-compartment variant ***HC*** retains both input pathways but shares a learnable decay factor $\lambda_l = \lambda_c$. As shown in Figure 5, CoSpAC performs best in both settings, followed by the two-compartment variant with equal decay factors and then the single-compartment variant. The improvement of HC over SC supports augmenting local observations with teammate messages through an additional dendritic compartment. CoSpAC further improves on HC, supporting the benefit of independently learned temporal integration timescales for the two input streams. Together, these results support

combining inter-agent communication with heterogeneous dendritic integration for asynchronous macro-action coordination.

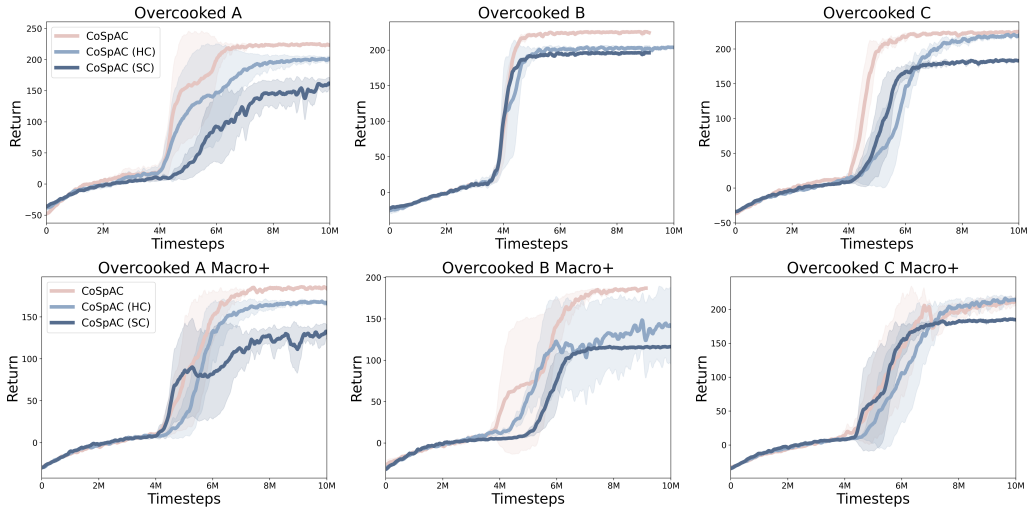

Figure 5: Ablation results under standard and heterogeneous macro-action durations. CoSpAC outperforms HC with shared decay and SC without teammate messages.

5.5 Energy Efficiency

We estimate the inference compute energy of CoSpAC and baselines using 3.97 pJ per multiply-accumulate operation and 77 fJ per synaptic operation, following Xu et al. (2026a). We use four internal time steps and an approximate firing rate of 31.7% for the spike layers. Table 1 shows lower estimated inference compute energy for CoSpAC than for the recurrent ANN baselines. Beyond this energy comparison, CoSpAC achieves higher sample efficiency and final returns on BoxPushing in Figure 2, supporting the use of compartmental spiking dynamics for asynchronous cooperation.

Table 1: Estimated inference compute energy (nJ) per policy–value evaluation pair.

Environment	Mac-IAICC	Mac-NIACC	ACAC	CoSpAC
BoxPushing	110.91	110.91	175.95	40.51
Overcooked	193.10	193.10	1054.94	59.63

6 Conclusion

In this work, we presented CoSpAC, a compartmentalized spiking actor-critic framework for asynchronous multi-agent reinforcement learning. Inspired by temporal dendritic heterogeneity, CoSpAC processes current local observations and available teammate messages through separate dendritic compartments with distinct learnable decay factors, allowing the two input streams to evolve at different temporal integration timescales before somatic integration for action selection. Communication follows agents' individual decision events, while a receiver-oriented objective trains message generation to support receiving agents' decisions. Experiments on BoxPushing and Overcooked demonstrate competitive or improved returns and sample efficiency compared with strong asynchronous MARL baselines. These results support compartmentalized temporal processing within spiking actors as an effective mechanism for coordinating agents with asynchronous decision times.

REFERENCES

Christopher Amato, George Konidaris, Gabriel Cruz, Christopher A Maynor, Jonathan P How, and Leslie P Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 1241–1248. IEEE, 2015.

Christopher Amato, George Konidaris, Leslie P Kaelbling, and Jonathan P How. Modeling and planning with macro-actions in decentralized pomdps. *Journal of Artificial Intelligence Research*, 64:817–859, 2019.

Romain Brette and Wulfram Gerstner. Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *Journal of neurophysiology*, 94(5):3637–3642, 2005.

Rishidev Chaudhuri, Kenneth Knoblauch, Marie-Alice Gariel, Henry Kennedy, and Xiao-Jing Wang. A large-scale circuit mechanism for hierarchical dynamical processing in the primate cortex. *Neuron*, 88(2):419–431, 2015.

Sydney Dolan, Siddharth Nayak, Jasmine Jerry Aloor, and Hamsa Balakrishnan. Asynchronous cooperative multi-agent reinforcement learning with limited communication. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '25, pp. 2496–2498, 2025.

Haowen Dou, Lujuan Dang, Mingfei Lu, Liangjun Chen, and Badong Chen. Efficient heterogeneous exploration with mutual policy divergence maximization for multiagent reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Shibo Feng, Wanjin Feng, Xingyu Gao, Peilin Zhao, and Zhiqi Shen. Ts-lif: a temporal segment spiking neuron network for time series forecasting. In *International Conference on Learning Representations*, volume 2025, pp. 28064–28084, 2025.

Mehmet Fişek, Dustin Herrmann, Alexander Egea-Weiss, Matilda Cloves, Lisa Bauer, Tai-Ying Lee, Lloyd E Russell, and Michael Häusser. Cortico-cortical feedback engages active dendrites in visual cortex. *Nature*, 617(7962):769–776, 2023.

Trong Nghia Hoang, Yuchen Xiao, Kavinayan Sivakumar, Christopher Amato, and Jonathan P Howl. Near-optimal adversarial policy switching for decentralized asynchronous multi-agent systems. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6373–6380. IEEE, 2018.

Whiyoung Jung, Sunghoon Hong, Deunsol Yoon, Kanghoon Lee, and Woohyung Lim. Agent-centric actor-critic for asynchronous multi-agent reinforcement learning. In *Forty-second International Conference on Machine Learning*, 2025.

Xiyun Li, Ziyi Ni, Jingqing Ruan, Linghui Meng, Jing Shi, Tielin Zhang, and Bo Xu. Mixture of personality improved spiking actor network for efficient multi-agent cooperation. *Frontiers in Neuroscience*, 17:1219405, 2023.

Xingyue Liang, Qiaoyun Wu, Yun Zhou, Chunyu Tan, Hongfu Yin, and Changyin Sun. Spiking reinforcement learning enhanced by bioinspired event source of multi-dendrite spiking neuron and dynamic thresholds. *IEEE/CAA Journal of Automatica Sinica*, 12(3):618–629, 2025.

Jingyang Ma, Chongming Liu, Songting Li, and Douglas Zhou. Beyond linear processing: Dendritic bilinear integration in spiking neural networks. In *International Conference on Learning Representations*, volume 2026, pp. 77297–77323, 2026.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Baskin Şenbaşlar and Gaurav S Sukhatme. Asynchronous real-time decentralized multi-robot trajectory planning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9972–9979. IEEE, 2022.

Junjie Sheng, Xiangfeng Wang, Bo Jin, Junchi Yan, Wenhao Li, Tsung-Hui Chang, Jun Wang, and Hongyuan Zha. Learning structured communication for multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 36(2):50, 2022.

Yinqian Sun, Feifei Zhao, Mingyang Lyu, and Yi Zeng. Spiking world model with multicompartment neurons for model-based reinforcement learning. *Proceedings of the National Academy of Sciences*, 122(50):e2513319122, 2025a.

Yinqian Sun, Feifei Zhao, Zhuoya Zhao, and Yi Zeng. Multi-compartment neuron and population encoding powered spiking neural network for deep distributional reinforcement learning. *Neural Networks*, 182:106898, 2025b.

Korneel Van den Berghe, Stein Stroobants, Vijay Janapa Reddi, and Guido De Croon. Adaptive surrogate gradients for sequential reinforcement learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 38:147904–147926, 2026.

Guokang Wang, Hang Li, Shuyuan Zhang, Di Guo, Yanhong Liu, and Huaping Liu. Observe then act: Asynchronous active vision-action model for robotic manipulation. *IEEE Robotics and Automation Letters*, 10(4):3422–3429, 2025.

Sarah A Wu, Rose E Wang, James A Evans, Joshua B Tenenbaum, David C Parkes, and Max Kleiman-Weiner. Too many cooks: Coordinating multi-agent collaboration through inverse planning. In *Proceedings of the annual meeting of the cognitive science society*, volume 42, 2020.

Yuchen Xiao, Joshua Hoffman, and Christopher Amato. Macro-action-based deep multi-agent reinforcement learning. In *Conference on Robot Learning*, pp. 1146–1161. PMLR, 2020a.

Yuchen Xiao, Joshua Hoffman, Tian Xia, and Christopher Amato. Learning multi-robot decentralized macro-action-based policies via a centralized q-net. In *2020 IEEE International conference on robotics and automation (ICRA)*, pp. 10695–10701. IEEE, 2020b.

Yuchen Xiao, Weihao Tan, and Christopher Amato. Asynchronous actor-critic for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 35:4385–4400, 2022.

Yuchen Xiao, Weihao Tan, Joshua Hoffman, Tian Xia, and Christopher Amato. Asynchronous multi-agent deep reinforcement learning under partial observability. *The International Journal of Robotics Research*, 44(8):1257–1286, 2025.

Zijie Xu, Tong Bu, Zecheng Hao, Jianhao Ding, and Zhaofei Yu. Proxy target: Bridging the gap between discrete spiking neural networks and continuous control. *Advances in Neural Information Processing Systems*, 38:159158–159184, 2026a.

Zijie Xu, Xinyu Shi, Yiting Dong, Zihan Huang, and Zhaofei Yu. Care-bn: Precise moving statistics for stabilizing spiking neural networks in reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 50749–50772, 2026b.

Zhile Yang, Shangqi Guo, Ying Fang, Zhaofei Yu, and Jian K Liu. Spiking variational policy gradient for brain inspired reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):1975–1990, 2024.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems*, 35:24611–24624, 2022.

Chao Yu, Xinyi Yang, Jiaxuan Gao, Jiayu Chen, Yunfei Li, Jijia Liu, Yunfei Xiang, Ruixin Huang, Huazhong Yang, Yi Wu, and Yu Wang. Asynchronous multi-agent reinforcement learning for efficient real-time multi-robot cooperative exploration. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '23, pp. 1107–1115, 2023.

Mengwen Yuan, Xi Wu, Rui Yan, and Huajin Tang. Reinforcement learning in spiking neural networks with stochastic and deterministic synapses. *Neural computation*, 31(12):2368–2389, 2019.

Yufeng Yuan and A Rupam Mahmood. Asynchronous reinforcement learning for real-time control of physical robots. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 5546–5552. IEEE, 2022.

Geng Zhang, Shuangming Yang, Xuetao Zhang, and Badong Chen. Biologically plausible unsupervised learning for self-organizing spiking neural networks with dendritic computation. *Neurocomputing*, 631:129707, 2025.

Han Zhang, Xiaohui Zhang, Zhao Feng, and Xiaohui Xiao. Heterogeneous multi-robot cooperation with asynchronous multi-agent reinforcement learning. *IEEE Robotics and Automation Letters*, 9 (1):159–166, 2023.

Shimin Zhang, Qu Yang, Chenxiang Ma, Jibin Wu, Haizhou Li, and Kay Chen Tan. Tc-lif: A two-compartment spiking neuron model for long-term sequential modelling. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 16838–16847, 2024.

Wenjing Zhang and Wei Zhang. Tomacvf: Temporal macro-action value factorization for asynchronous multi-agent reinforcement learning. In *ECAI 2025: 28th European Conference on Artificial Intelligence 25-30 October 2025, Bologna, Italy-Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025) Proceedings*, pp. 3799–3806. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, 2025.

Hanle Zheng, Zhong Zheng, Rui Hu, Bo Xiao, Yujie Wu, Fangwen Yu, Xue Liu, Guoqi Li, and Lei Deng. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15(1):277, 2024.

A APPENDIX

A.1 THE ALGORITHM OF COSPAC

Algorithm 1 summarizes the training procedure of CoSpAC. Each iteration collects asynchronous trajectories at agents' individual decision times. At each decision event, the actor retrieves and aggregates available teammate messages, processes them alongside local observations through separate dendritic dynamics, and integrates the resulting states at the soma for action selection. Messages are also generated at these decision events for use by other agents. The centralized critic provides value estimates for computing advantages, after which the actor, critic, and message parameters are updated using the PPO, value regression, and receiver-oriented objectives, respectively. The dendritic decay parameters are optimized together with the actor parameters.

Algorithm 1 Compartmentalized Spiking Actor-Critic

Require: Actor parameters θ (including decay parameters α_l, α_c), critic parameters ϕ, and message parameters ψ

for each training iteration **do**
  Collect asynchronous trajectories: $H^i = \{(x_k^i, a_k^i, r_k^i, \tau_k^i)\}_{k=1}^{K_i}$
  for each decision event (i, τ_k^i) **do**
    Retrieve the latest available message from each teammate via $\mathcal{M}_i(\tau_k^i) = \{\hat{m}^j(\tau_k^i) \mid j \neq i\}$
    Aggregate teammate information via $\tilde{m}_k^i = Agg(\mathcal{M}_i(\tau_k^i))$
    for $q = 1, \ldots, Q$ **do**
      Update dendritic states via $h_{k,q}^{i,l} = \lambda_l h_{k,q-1}^{i,l} + W_l x_k^i$ and $h_{k,q}^{i,c} = \lambda_c h_{k,q-1}^{i,c} + W_c \tilde{m}_k^i$
      Somatic integration via $I_{k,q}^{i,s} = W_s[h_{k,q}^{i,l}; h_{k,q}^{i,c}]$ and $s_{k,q}^{i,s} = \Theta(\tilde{h}_{k,q}^{i,s} - \vartheta_s)$
    end for
    Message generation via $m_k^i = g_\psi(h_{k,Q}^{i,l}, \tau_k^i)$
  end for
  Compute advantages via $V_{\phi,k} = V_\phi\left(H^{1:N}; \tau_k^i\right)$, $A_k^i = \hat{A}(H^{1:N}, \tau_k^i)$ and MacroGAE (Jung et al., 2025)
  Compute $\mathcal{L}_{\text{actor}}^{\theta^{1:n}} = \mathcal{L}_{\text{PPO}}^{\theta^{1:n}}$, $\mathcal{L}_V^\phi = \mathbb{E}_k\left[\left(V_{\phi,k} - \hat{V}_k\right)^2\right]$ and $\mathcal{L}_{\text{msg}}^\psi$, and update θ, ϕ, ψ.
end for
return θ, ϕ, ψ

A.2 DETAILED EXPERIMENTAL SETTINGS

All experiments are conducted on an NVIDIA GeForce RTX 3090 GPU and an Intel Xeon CPU. Each experiment uses three random seeds. Learning curves show the mean return across seeds, with shaded regions indicating one standard deviation. Table 2 summarizes the main PPO and SNN hyperparameters of CoSpAC.

A.3 ADDITIONAL RESULTS

We further evaluate CoSpAC on Overcooked A-C with macro-action duration multipliers sampled separately for each random seed. At the start of each training run, the multipliers for the first two agents are sampled from $[0.6, 0.8]$ and $[0.8, 1.0]$, respectively, while the third agent's multiplier is set to 1. These multipliers scale the nominal action durations and remain fixed throughout the run. A new pair is sampled for the next random seed, allowing us to evaluate performance across different execution-time configurations.

Fig. 6 shows that CoSpAC achieves higher mean returns than ACAC across all three layouts, consistent with the *Macro+* results in the main text. Additionally, Fig. 7 shows the same performance comparison among the model variants, with CoSpAC achieving the highest mean returns, followed by the homogeneous two-compartment variant (HC) and the single-compartment variant (SC). The improvement over HC supports the benefit of independently learned temporal integration timescales for local observations and teammate messages. These results suggest that CoSpAC is more robust to

Table 2: PPO and SNN hyperparameters by environment.

Hyperparameter	BoxPushing	Overcooked	Overcooked-Rand
PPO training			
Discount factor γ	0.99	0.99	0.99
GAE coefficient	0.95	0.95	0.95
PPO clipping parameter	0.05	0.01	0.01
Initial actor learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}
Critic learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}
Initial message learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}
Actor learning rate decay	Linear	Linear	Linear
Final actor learning rate	1×10^{-5}	3×10^{-5}	3×10^{-5}
PPO epochs	5	5	5
Value loss coefficient	1.0	1.0	1.0
Entropy coefficient	0.01	0.01	0.01
Parallel environments	8	8	8
SNN training			
Internal time steps Q	2	4	4
Actor hidden dimension	32	32	32
Critic hidden dimension	32	64	64
Actor firing threshold	1.0	1.0	1.0
Critic LIF decay parameter	0.5	0.5	0.5
Critic firing threshold	0.5	0.5	0.5
Message dimension	16	16	16

differences in agents' execution durations, supporting heterogeneous temporal integration for asynchronous coordination.

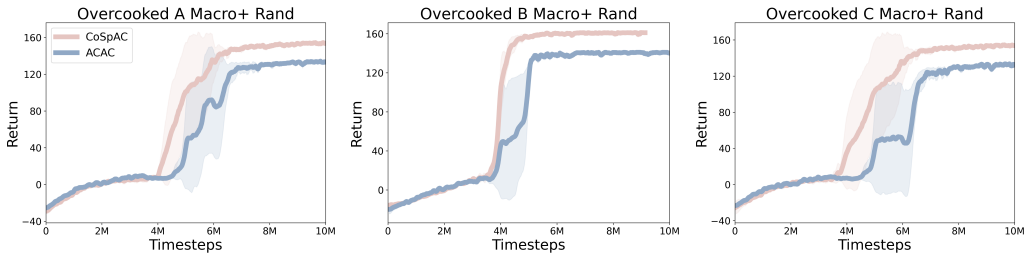

Figure 6: CoSpAC and ACAC on Overcooked A-C with duration multipliers sampled once per training run and held fixed throughout. CoSpAC achieves higher mean returns across all three layouts.

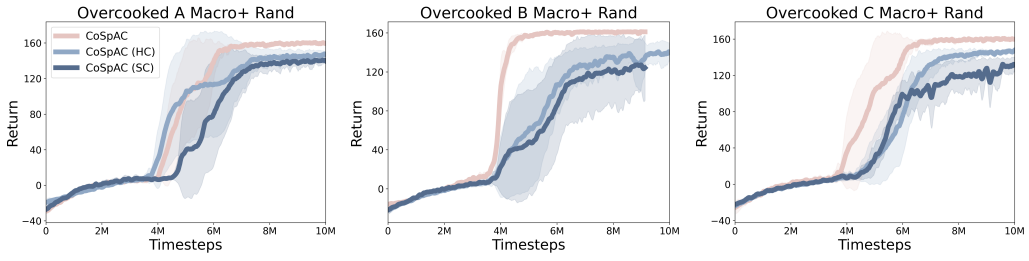

Figure 7: Ablation results on Overcooked A-C with duration multipliers sampled once per training run and held fixed throughout. CoSpAC achieves the highest mean returns, followed by HC and SC.

A.4 LIMITATIONS

CoSpAC learns distinct dendritic decay factors for local observations and teammate messages. These factors remain fixed during inference and do not explicitly adapt to the age of individual messages. Allowing temporal integration to adjust to message age is a possible direction for future work.

LLM USE STATEMENT

Large language models were used to assist with language editing, manuscript organization, and discussions of methods, experimental design, and result interpretation. The authors retain responsibility for the content and conclusions of this work.