Attention-Based Recurrence for Multi-Agent Reinforcement Learning under State Uncertainty

Thomy Phan
LMU Munich
thomy.phan@ifi.lmu.de

Fabian Ritz
LMU Munich

Jonas Nüßlein
LMU Munich

Michael Kölle
LMU Munich

Thomas Gabor
LMU Munich

Claudia Linnhoff-Popien
LMU Munich

ABSTRACT

State uncertainty poses a major challenge for decentralized coordination but is largely neglected in state-of-the-art research due to a strong focus on state-based *centralized training for decentralized execution (CTDE)* and benchmarks that lack sufficient stochasticity like *StarCraft Multi-Agent Challenge (SMAC)*. In this paper, we propose *Attention-based Embeddings of Recurrence In multi-Agent Learning (AERIAL)* to approximate value functions under agent-wise state uncertainty. AERIAL replaces the true state with a learned representation of multi-agent recurrence, considering more accurate information about decentralized agent decisions than state-based CTDE. We then introduce *MessySMAC*, a modified version of SMAC with stochastic observations and higher variance in initial states, to provide a more general and configurable benchmark regarding state uncertainty. We evaluate AERIAL in Dec-Tiger as well as in a variety of SMAC and MessySMAC maps, and compare the results with state-based CTDE. Furthermore, we evaluate the robustness of AERIAL and state-based CTDE against various state uncertainty configurations in MessySMAC.

KEYWORDS

Dec-POMDP, State Uncertainty, Multi-Agent Learning, Recurrence, Self-Attention

1 INTRODUCTION

A wide range of real-world applications like fleet management, industry 4.0, or communication networks can be formulated as *decentralized partially observable Markov decision process (Dec-POMDP)* representing a cooperative *multi-agent system (MAS)*, where agents have to coordinate in a decentralized way to achieve a common goal [8, 21]. *State uncertainty* poses a major challenge for decentralized coordination in Dec-POMDPs due to noisy sensors and potentially high variance in initial states which are common in the real world [15, 21].

Multi-agent reinforcement learning (MARL) is a general approach to tackle Dec-POMDPs with remarkable progress in recent years [33, 35]. State-of-the-art MARL is based on *centralized training for decentralized execution (CTDE)*, where training takes place in a laboratory or a simulator with access to global information [8, 17]. For example, *state-based CTDE* exploits true state information to learn a centralized value function in order to derive coordinated policies for decentralized decision making [25, 36]. Due to its effectiveness in the *StarCraft Multi-Agent Challenge (SMAC)* as the current de facto standard for MARL evaluation, state-based CTDE has become very popular and is widely considered an adequate approach to general Dec-POMDPs, leading to many increasingly complex algorithms [18, 19].

However, merely relying on state-based CTDE and SMAC in MARL research can be a pitfall in practice as state uncertainty is largely neglected – despite being an important aspect in Dec-POMDPs [18]:

From an *algorithm perspective*, purely state-based value functions are insufficient to evaluate and adapt multi-agent behavior, since all agents make decisions on a completely different basis, i.e., individual histories of noisy observations and actions. True Dec-POMDP value functions consider more accurate closed-loop information about decentralized agent decisions though [22]. Furthermore, the optimal state-based value function represents an upper-bound of the true optimal Dec-POMDP value function thus state-based CTDE can result in overly optimistic behavior in general Dec-POMDPs [18].

From a *benchmark perspective*, SMAC has very limited state uncertainty due to deterministic observations and low variance in initial states [6]. Therefore, SMAC scenarios only represent simplified special cases rather than general Dec-POMDP challenges, being insufficient for evaluating generality of MARL [18].

In this paper, we propose *Attention-based Embeddings of Recurrence In multi-Agent Learning (AERIAL)* to approximate value functions under agent-wise state uncertainty. AERIAL replaces the true state with a learned representation of multi-agent recurrence, considering more accurate closed-loop information about decentralized agent decisions than state-based CTDE. We then introduce *MessySMAC*, a modified version of SMAC with stochastic observations and higher variance in initial states, to provide a more general and configurable Dec-POMDP benchmark for more adequate evaluation under state uncertainty.

Our contributions are as follows:

- We formulate and discuss the concepts of AERIAL w.r.t. state uncertainty in general Dec-POMDPs.
- We introduce MessySMAC to enable systematic evaluation under various state uncertainty configurations.
- We evaluate AERIAL in Dec-Tiger, a small and traditional Dec-POMDP benchmark, as well as in a variety of original SMAC and MessySMAC maps, and compare the results with state-based CTDE. Our results show that AERIAL achieves competitive performance in original SMAC, and superior performance in Dec-Tiger and MessySMAC. Furthermore, we evaluate the robustness of AERIAL and state-based CTDE against various state uncertainty configurations in MessySMAC.

2 BACKGROUND

2.1 Decentralized POMDPs

We formulate cooperative MAS problems as *Dec-POMDP* $M = \langle \mathcal{D}, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{Z}, \Omega, b_0 \rangle$, where $\mathcal{D} = \{1, ..., N\}$ is a set of agents i, $\mathcal{S}$ is a set of (true) states s_t at time step t, $\mathcal{A} = \langle \mathcal{A}_i \rangle_{i\in\mathcal{D}}$ is the set of joint actions $\mathbf{a_t} = \langle a_{t,1}, ..., a_{t,N} \rangle = \langle a_{t,i} \rangle_{i\in\mathcal{D}}$, $\mathcal{T}(s_{t+1}|s_t, \mathbf{a_t})$ is the state transition probability, $r_t = \mathcal{R}(s_t, \mathbf{a_t}) \in \mathbb{R}$ is the shared reward, $\mathcal{Z}$ is a set of local observations $z_{t,i}$ for each agent $i \in \mathcal{D}$, $\Omega(\mathbf{z_{t+1}}|\mathbf{a_t}, s_{t+1})$ is the probability of joint observation $\mathbf{z_{t+1}} = \langle z_{t+1,i} \rangle_{i\in\mathcal{D}} \in \mathcal{Z}^N$, and b_0 is the probability distribution over initial states s_0 [21]. Each agent i maintains a *local history* $\tau_{t,i} \in (\mathcal{Z}\times\mathcal{A}_i)^t$ and $\boldsymbol{\tau_t} = \langle \tau_{t,i} \rangle_{i\in\mathcal{D}}$ is the *joint history*. A *belief state* $b(s_t|\boldsymbol{\tau_t})$ is a sufficient statistic for joint history $\boldsymbol{\tau_t}$ and defines a probability distribution over true states s_t, updatable by Bayes' theorem [15]. Joint quantities are written in bold face.

State uncertainty in M is given by *observation stochasticity* w.r.t. Ω and *initialization stochasticity* w.r.t. b_0.

A *joint policy* $\boldsymbol{\pi} = \langle \pi_i \rangle_{i\in\mathcal{D}}$ with *decentralized* or *local policies* π_i defines a deterministic mapping from joint histories to joint actions $\boldsymbol{\pi}(\boldsymbol{\tau_t}) = \langle \pi_i(\tau_{t,i}) \rangle_{i\in\mathcal{D}} \in \mathcal{A}$. The *return* is defined by $G_t = \sum_{c=0}^{T-1} \gamma^c r_{t+c}$, where T is the *horizon* and $\gamma \in [0, 1]$ is the *discount factor*. $\boldsymbol{\pi}$ can be evaluated with a *value function* $Q^{\boldsymbol{\pi}}(\boldsymbol{\tau_t}, \mathbf{a_t}) = \mathbb{E}_{b_0,\mathcal{T},\Omega}[G_t|\boldsymbol{\tau_t}, \mathbf{a_t}, \boldsymbol{\pi}]$. The goal is to find an *optimal joint policy* $\boldsymbol{\pi}^*$ with *optimal value function* $Q^{\boldsymbol{\pi}^*} = Q^*$ as defined in the next section.

2.2 Optimal Value Functions and Policies

Fully Observable MAS. In MDP-like settings with a centralized controller, the optimal value function Q^*_{MDP} is defined by [3, 34]:

$$Q^*_{MDP}(s_t, \mathbf{a_t}) = r_t + \gamma \sum_{s_{t+1}\in\mathcal{S}} \mathcal{X} \qquad (1)$$

where $\mathcal{X} = \mathcal{T}(s_{t+1}|s_t, \mathbf{a_t}) max_{\mathbf{a_{t+1}}\in\mathcal{A}} Q^*_{MDP}(s_{t+1}, \mathbf{a_{t+1}})$.

Due to full observability, Q^*_{MDP} does not depend on $\boldsymbol{\tau_t}$ but on s_t. Thus, decentralized observations $z_{t,i}$ and probabilities according to Ω and b_0 are not considered at all. An optimal (joint) policy $\boldsymbol{\pi}^*_{\mathbf{MDP}}$ of the centralized controller simply maximizes Q^*_{MDP} for all s_t [34]:

$$\boldsymbol{\pi}^*_{\mathbf{MDP}} = argmax_{\boldsymbol{\pi}_{\mathbf{MDP}}} \sum_{s_t\in\mathcal{S}} Q^*_{MDP}(s_t, \boldsymbol{\pi}_{\mathbf{MDP}}(s_t)) \qquad (2)$$

Partially Observable MAS. In Dec-POMDPs, where true states are not fully observable and only decentralized controllers or agents exist, the optimal value function Q^* is defined by [22]:

$$Q^*(\boldsymbol{\tau_t}, \mathbf{a_t}) = \sum_{s_t\in\mathcal{S}} b(s_t|\boldsymbol{\tau_t}) \left(r_t + \gamma \sum_{s_{t+1}\in\mathcal{S}} \sum_{\mathbf{z_{t+1}}\in\mathcal{Z}^N} \mathcal{X} \right) \qquad (3)$$

where $\mathcal{X} = \mathcal{T}(s_{t+1}|s_t, \mathbf{a_t})\Omega(\mathbf{z_{t+1}}|\mathbf{a_t}, s_{t+1})Q^*(\boldsymbol{\tau_{t+1}}, \boldsymbol{\pi}^*(\boldsymbol{\tau_{t+1}}))$ with $\boldsymbol{\tau_{t+1}}$ consisting of $\boldsymbol{\tau_t}$, $\mathbf{a_t}$, and $\mathbf{z_{t+1}}$.

An optimal joint policy $\boldsymbol{\pi}^*$ for decentralized execution maximizes the expectation of Q^* for all joint histories $\boldsymbol{\tau_t}$ [7, 22]:

$$\boldsymbol{\pi}^* = argmax_{\boldsymbol{\pi}} \sum_{t=0}^{T-1} \sum_{\boldsymbol{\tau_t}\in(\mathcal{Z}^N\times\mathcal{A})^t} C^{\boldsymbol{\pi}}(\boldsymbol{\tau_t})\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau_t}|b_0)Q^*(\cdot) \qquad (4)$$

where $Q^*(\cdot) = Q^*(\boldsymbol{\tau_t}, \boldsymbol{\pi}(\boldsymbol{\tau_t}))$, indicator $C^{\boldsymbol{\pi}}(\boldsymbol{\tau_t})$ filters out joint histories $\boldsymbol{\tau_t}$ that are inconsistent with $\boldsymbol{\pi}$, and probability $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau_t}|b_0)$ represents the *recurrence* of all agents considering agent-wise state uncertainty w.r.t. decentralization of $\boldsymbol{\pi}$ and $\boldsymbol{\tau_t}$ [22]:

$$\begin{aligned} \mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau_t}|b_0) &= \mathbf{P}(\mathbf{z_0}|b_0) \prod_{c=1}^{t} \mathbf{P}(\mathbf{z_c}|\boldsymbol{\tau_{c-1}}, \boldsymbol{\pi}) \\ &= \mathbf{P}(\mathbf{z_0}|b_0) \prod_{c=1}^{t} \sum_{s_c\in\mathcal{S}} \sum_{s_{c-1}\in\mathcal{S}} \mathcal{T}(\cdot)\Omega(\cdot) \end{aligned} \qquad (5)$$

where $\mathcal{T}(\cdot) = \mathcal{T}(s_c|s_{c-1}, \boldsymbol{\pi}(\boldsymbol{\tau_{c-1}}))$ and $\Omega(\cdot) = \Omega(\mathbf{z_c}|\boldsymbol{\pi}(\boldsymbol{\tau_{c-1}}), s_c)$.

Since all agents act according to their local history $\tau_{t,i}$ without access to the complete joint history $\boldsymbol{\tau_t}$, recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau_t}|b_0)$ depends on more accurate *closed-loop information* than just true states s_t, i.e., all previous observations, actions, and probabilities according to b_0, $\mathcal{T}$, and Ω.

Q^*_{MDP} is proven to represent an upper bound of Q^* [22]. Thus, deriving local policies π_i from Q^*_{MDP} instead of Q^* can result in overly optimistic behavior as we will show in Section 4.1 and 6.

2.3 Multi-Agent Reinforcement Learning

Finding an optimal joint policy $\boldsymbol{\pi}^*$ via exhaustive computation of Q^* according to Eq. 3-5 is intractable in practice [20, 29]. MARL offers a scalable way to learn Q^* and $\boldsymbol{\pi}^*$ via function approximation, e.g., using CTDE, where training takes place in a laboratory or a simulator with access to global information [8, 17]. We focus on value-based MARL to learn a centralized value function $Q_{tot} \approx Q^*$, which can be factorized into *local utility functions* $\langle Q_i \rangle_{i\in\mathcal{D}}$ for decentralized decision making via $\pi_i(\tau_{t,i}) = argmax_{a_{t,i}\in\mathcal{A}_i} Q_i(\tau_{t,i}, a_{t,i})$. For that, a *factorization operator* Ψ is used [23]:

$$Q_{tot}(\boldsymbol{\tau_t}, \mathbf{a_t}) = \Psi(Q_1(\tau_{t,1}, a_{t,1}), ..., Q_N(\tau_{t,N}, a_{t,N})) \qquad (6)$$

In practice, Ψ is realized with deep neural networks, such that $\langle Q_i \rangle_{i\in\mathcal{D}}$ can be learned end-to-end via backpropagation by minimizing the mean squared *temporal difference (TD)* error [25, 28]. A factorization operator Ψ is *decentralizable* when satisfying the *IGM (Individual-Global-Max)* such that [27]:

$$argmax_{\mathbf{a_t}\in\mathcal{A}} Q_{tot}(\boldsymbol{\tau_t}, \mathbf{a_t}) = \langle argmax_{a_{t,i}\in\mathcal{A}_i} Q_i(\tau_{t,i}, a_{t,i}) \rangle_{i\in\mathcal{D}} \qquad (7)$$

There exists a variety of factorization operators Ψ, which satisfy Eq. 7 using monotonicity like QMIX [25], nonlinear transformation like QPLEX [33], or loss weighting like CW- and OW-QMIX [24]. Most approaches use state-based CTDE to learn Q^*_{MDP} according to Eq. 1 instead of Q^* (Eq. 3-5).

2.4 Recurrent Reinforcement Learning

In partially observable settings, the policy π_i of agent i conditions on the history $\tau_{t,i}$ of past observations and actions [15, 21]. In practice, *recurrent neural networks (RNNs)* like LSTMs or GRUs are used to learn a compact representation $h_{t,i}$ of $\tau_{t,i}$ and π_i known as *hidden state* or *memory representation*[1], which implicitly encodes the *individual recurrence* of agent i, i.e., the distribution $P_i^{\pi_i}$ over $\tau_{t,i}$ [5, 11, 12]:

$$P_i^{\pi_i}(\tau_{t,i}|b_0) = P_i(z_{0,i}|b_0) \prod_{c=1}^{t} P_i(z_{c,i}|\tau_{c-1,i}, \pi_i) \qquad (8)$$

[1] In this paper, we use the term *memory representation* to avoid confusion with the state terminology of the (Dec-)POMDP literature [15, 21].

RNNs are commonly used for partially observable problems and have been empirically shown to be more effective than using raw observations $z_{t,i}$ or histories $\tau_{t,i}$ [10, 26, 32].

3 RELATED WORK

Multi-Agent Reinforcement Learning. In recent years, MARL has achieved remarkable progress in challenging domains [9, 32]. State-of-the-art MARL is based on CTDE to learn a centralized value function Q_{tot} for actor-critic learning [8, 17, 36] or factorization [24, 25, 33]. However, the majority of works assumes a simplified Dec-POMDP setting, where Ω is deterministic, and uses true states to approximate Q^*_{MDP} according to Eq. 1 instead of Q^* (Eq. 3-5). Thus, state-based CTDE is possibly less effective in more general Dec-POMDP settings due to neglecting important closed-loop information about decentralized agent decisions [6, 18]. Our approach addresses state uncertainty by using a *learned representation* of recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_t|b_0)$ according to Eq. 5 instead of true states s_t.

Weaknesses of State-Based CTDE. Recent works investigated potential weaknesses of state-based CTDE for multi-agent actor-critic methods regarding bias and variance [18, 19]. The experimental results show that state-based CTDE can surprisingly fail in very simple Dec-POMDP benchmarks that exhibit more state uncertainty than SMAC. While these studies can be considered an important step towards general Dec-POMDPs, there is neither an approach which adequately addresses state uncertainty nor a benchmark to systematically evaluate such an approach yet. In this work, we focus on *value-based MARL*, where learning an accurate value function is important for meaningful factorization, and propose an *attention-based recurrence approach* to value function approximation under state uncertainty. We also introduce a *modified* SMAC benchmark, which enables systematic evaluation under various state uncertainty configurations.

Attention-Based CTDE. Attention has been used in CTDE to process information of potentially variable length N, where joint observations $\mathbf{z_t}$, joint actions $\mathbf{a_t}$, or local utilities $\langle Q_i \rangle_{i\in\mathcal{D}}$ are weighted and aggregated to provide a meaningful representation for value function approximation [13, 14, 16, 33, 35]. Most works focus on Markov games without observation stochasticity, which are special cases of the Dec-POMDP setting. In this work, we focus on *state uncertainty* and apply *self-attention* to the *memory representations* $h_{t,i}$ of all agents' RNNs instead of the raw observations $z_{t,i}$ to approximate Q^* for *general Dec-POMDPs* according to Eq. 3-5.

4 AERIAL

4.1 Limitation of State-Based CTDE

Most state-of-the-art works assume a simplified Dec-POMDP setting, where Ω is deterministic, and approximate Q^*_{MDP} according to Eq. 1 instead of Q^* (Eq. 3-5).

If there are only deterministic observations and initial states s_0 such that $b_0(s_0) = 1$ and $b_0(s') = 0$ if $s' \neq s_0$, then recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_t|b_0)$ as defined in Eq. 5 would only depend on state transition probabilities $\mathcal{T}(s_{t+1}|s_t, \mathbf{a_t})$ which are purely state-based, ignoring decentralization of agents and observations [22]. In such scenarios, state uncertainty is very limited, especially if all π_i are deterministic. We hypothesize that this is one reason for the empirical success of state-based CTDE in original SMAC, whose scenarios seemingly have these simplifying properties [6, 18].

In the following, we regard a small example, where state-based CTDE can fail at finding an optimal joint policy $\boldsymbol{\pi}^*$.

Example. *Dec-Tiger* is a simple Dec-POMDP with $N = 2$ agents facing two doors [20]. A tiger is randomly placed behind the left (s_L) or right door (s_R) representing the true state. Both agents are able to listen (*li*) and open the left (o_L) or right door (o_R). The listening action *li* produces a noisy observation of either hearing the tiger to be left (z_L) or right (z_R), correctly indicating the tiger's position with 85% chance and a cost of -1 per listening agent. If both agents open the same door, the episode terminates with a reward of -50 if opening the tiger door and +20 otherwise. If both agents open different doors, the episode ends with -100 reward and if only one agent opens a door while the other agent is listening, the episode terminates with -101 if opening the tiger door and +9 otherwise.

Given a horizon of $T = 2$, the tiger being behind the right door (s_R), and both agents having listened in the first step, where agent 1 heard z_L and agent 2 heard z_R: Assuming that both agents learned to perform the same actions, e.g., due to CTDE and parameter sharing [9, 30], Q^*_{MDP} and Q^* would estimate the following values[2]:

$$\begin{array}{ll} Q^*_{MDP}(s_R, \langle li, li\rangle) = -2 & Q^*(\boldsymbol{\tau}_t, \langle li, li\rangle) = -2 \\ Q^*_{MDP}(s_R, \langle o_L, o_L\rangle) = 20 & Q^*(\boldsymbol{\tau}_t, \langle o_L, o_L\rangle) = -15 \\ Q^*_{MDP}(s_R, \langle o_R, o_R\rangle) = -50 & Q^*(\boldsymbol{\tau}_t, \langle o_R, o_R\rangle) = -15 \end{array}$$

Any policy $\boldsymbol{\pi}^*_{\mathbf{MDP}}$ or decentralizable joint policy $\boldsymbol{\pi}$ w.r.t. IGM (Eq. 7) that maximizes Q^*_{MDP} according to Eq. 2 would optimistically recommend $\langle o_L, o_L\rangle$ based on the true state s_R, regardless of what the agents actually observed. However, any joint policy $\boldsymbol{\pi}^*$ that maximizes the expectation of Q^* according to Eq. 4 would consider agent-wise state uncertainty and recommend $\langle li, li\rangle$ which corresponds to the true optimal decision for $T = 2$ [29].

4.2 Attention-Based Embeddings of Recurrence

Preliminaries. We now introduce *Attention-based Embeddings of Recurrence In multi-Agent Learning (AERIAL)* to approximate optimal Dec-POMDP value functions Q^* according to Eq. 3-5. Our setup uses a factorization operator Ψ like QMIX or QPLEX according to Eq. 6-7. All agents process their local histories $\tau_{t,i}$ via RNNs as motivated in Section 2.4 and schematically shown in Fig. 1 (left).

Unlike Q^*_{MDP}, the true optimal Dec-POMDP value function Q^* considers more accurate closed-loop information about decentralized agent decisions through recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_t|b_0)$ according to Eq. 5. Simply replacing s_t with $\boldsymbol{\tau}_t$ as suggested in [18] is not sufficient because the resulting value function would assume a centralized controller which has access to the complete joint history $\boldsymbol{\tau}_t$, in contrast to decentralized agents i which can only access their respective local history $\tau_{t,i}$ [22].

Exploiting Multi-Agent Recurrence. At first we propose to naively exploit all individual recurrences by simply replacing the true state s_t in CTDE with the *joint memory representation* $\mathbf{h_t} = \langle h_{t,i}\rangle_{i\in\mathcal{D}}$ of all agents' RNNs. Each memory representation $h_{t,i}$ implicitly encodes the individual recurrence $P_i^{\pi_i}(\tau_{t,i}|b_0)$ of agent i

[2] The exact calculation is provided in the appendix A.

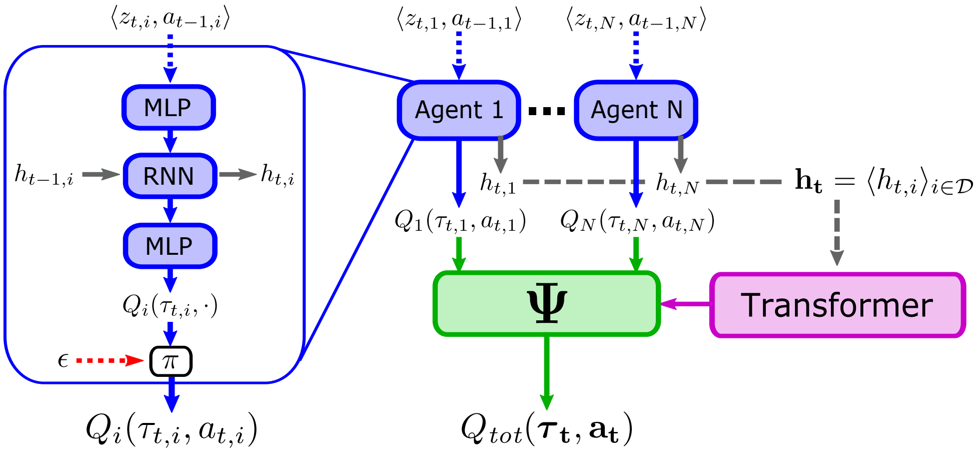

$\langle z_{t,i}, a_{t-1,i}\rangle$
MLP
$h_{t-1,i}$
RNN
$h_{t,i}$
MLP
$Q_i(\tau_{t,i}, \cdot)$
ϵ
π
$Q_i(\tau_{t,i}, a_{t,i})$
$\langle z_{t,1}, a_{t-1,1}\rangle$
$\langle z_{t,N}, a_{t-1,N}\rangle$
Agent 1
Agent N
$h_{t,1}$
$h_{t,N}$
$\mathbf{h_t} = \langle h_{t,i}\rangle_{i\in\mathcal{D}}$
$Q_1(\tau_{t,1}, a_{t,1})$
$Q_N(\tau_{t,N}, a_{t,N})$
Ψ
Transformer
$Q_{tot}(\boldsymbol{\tau}_\mathbf{t}, \mathbf{a_t})$

Figure 1: Illustration of the AERIAL setup. *Left:* Recurrent agent network structure with memory representations $h_{t-1,i}$ and $h_{t,i}$. *Right:* Value function factorization via factorization operator Ψ using the joint memory representation $\mathbf{h_t} = \langle h_{t,i}\rangle_{i\in\mathcal{D}}$ of all agents' RNNs instead of true states s_t. All memory representations $h_{t,i}$ are detached from the computation graph to avoid additional differentiation (indicated by the dashed gray arrows) and passed through a simplified transformer before being used by Ψ for value function factorization.

according to Eq. 8. Therefore, $\mathbf{h_t}$ provides more accurate closed-loop information about decentralized agent decisions than s_t.

This approach, called `AERIAL (no attention)`, can already be considered a sufficient solution if all individual recurrences $P_i^{\pi_i}(\tau_{t,i}|b_0)$ are conditionally independent such that $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_\mathbf{t}|b_0) = \prod_{i=1}^{N} P_i^{\pi_i}(\tau_{t,i}|b_0)$.

Attention-Based Recurrence. While `AERIAL (no attention)` offers a simple way to address agent-wise state uncertainty, the independence assumption of all individual recurrences $P_i^{\pi_i}(\tau_{t,i}|b_0)$ does not hold in practice due to correlations in observations and actions [1, 2]. Given the Dec-Tiger example above, the probability for being in state s_R is 0.15 and 0.85 from the perspective of agent 1 and 2 respectively [15]. However, the actual probability according to the belief state $b(s_R|\boldsymbol{\tau}_\mathbf{t})$ is $0.5 \neq 0.15 \cdot 0.85$, indicating that all $P_i^{\pi_i}(\tau_{t,i}|b_0)$ are not conditionally independent [21].

Therefore, we process $\mathbf{h_t}$ by a simplified *transformer* to automatically consider the latent dependencies of all memory representations $h_{t,i} \in \mathbf{h_t}$ through self-attention [31]. The resulting approach, called `AERIAL`, is depicted in Fig. 1 and Algorithm 1 in appendix B.

Our transformer does not use positional encoding or masking. The joint memory representation $\mathbf{h_t}$ is simply passed through a *multi-head attention* layer with the output of each attention head c being defined by [31]:

$$att_c(\mathbf{h_t}) = softmax\left(\frac{W_q^c(\mathbf{h_t})W_k^c(\mathbf{h_t})^\top}{\sqrt{d_{att}}}\right)W_v^c(\mathbf{h_t}) \tag{9}$$

where W_q^c, W_k^c, and W_v^c are *multi-layer perceptrons (MLP)* with an output dimensionality of d_{att}. All outputs $att_c(\mathbf{h_t})$ are summed and passed through a series of MLP layers before being input to the factorization operator Ψ, effectively replacing the true state s_t by a learned representation of recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_\mathbf{t}|b_0)$ according to Eq. 5.

To avoid additional differentation of $\mathbf{h_t}$ through Ψ or Eq. 9, we detach $\mathbf{h_t}$ from the computation graph. Thus, we make sure that $\mathbf{h_t}$ is only learned through agent RNNs.

4.3 Discussion of AERIAL

The strong focus on state-based CTDE in the last few years has led to many increasingly complex algorithms that largely neglect state uncertainty in general Dec-POMDPs [18]. In contrast, AERIAL offers a simple way to adjust factorization approaches by replacing the true state s_t with a learned representation of multi-agent recurrence to consider more accurate closed-loop information about decentralized agent decisions. The rest of the training scheme remains unchanged which eases adaptation.

Since the naive independence assumption of individual memory representations $h_{t,i}$ does not hold in practice despite the decentralization, we use a transformer to consider the latent dependencies of all $h_{t,i} \in \mathbf{h_t}$ to learn an adequate representation of recurrence $\mathbf{P}^{\boldsymbol{\pi}}(\boldsymbol{\tau}_\mathbf{t}|b_0)$ according to Eq. 5.

AERIAL does not depend on true states therefore requiring less overall information than state-based CTDE, since we assume $\mathbf{h_t}$ to be available in all CTDE setups anyway [8, 24]. Note that AERIAL does not necessarily require RNNs to obtain $\mathbf{h_t}$ as hidden layers of MLPs or decision transformers can be used as well to approximate $\mathbf{h_t}$ [4, 27].

5 MESSY SMAC

StarCraft Multi-Agent Challenge (SMAC) provides a rich set of micromanagement tasks, where a team of learning agents has to fight against an enemy team, which acts according to handcrafted heuristics of the built-in StarCraft AI [26]. SMAC currently represents the de facto standard for MARL evaluation [24, 25, 33]. However, SMAC scenarios exhibit very limited state uncertainty due to deterministic observations and low variance in initial states therefore

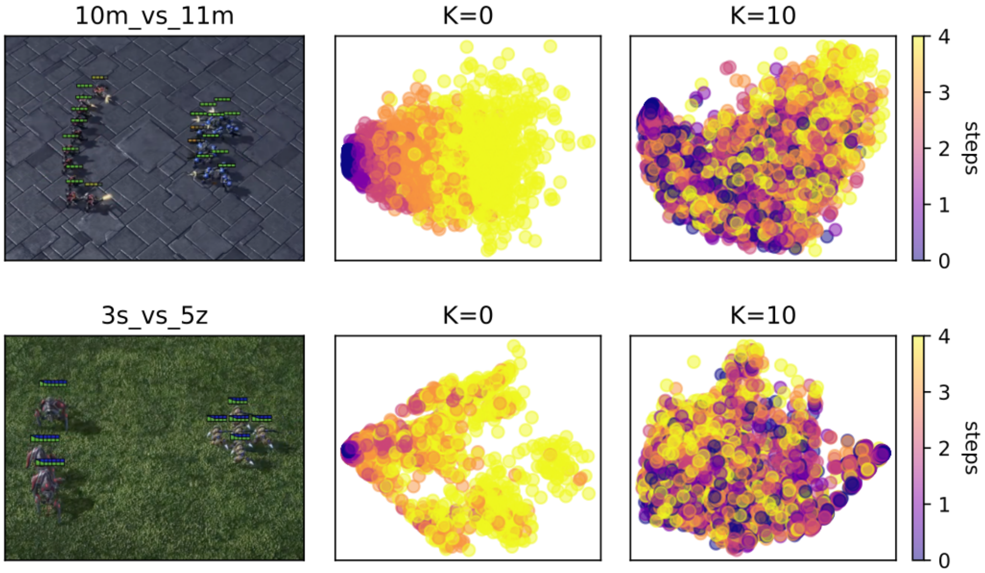


Figure 2: *Left:* Screenshot of two SMAC maps. *Middle:* PCA visualization of the joint observations in original SMAC within the first 5 steps of 1,000 episodes using a random policy (with $K = 0$ initial random steps). *Right:* Analogous PCA visualization for MessySMAC (with $K = 10$ initial random steps). For visual comparability, the observations are deterministic here.

only representing simplified special cases rather than general Dec-POMDP challenges [6, 18]. To assess practicability of MARL, we need benchmarks with sufficient state uncertainty as the real-world is generally messy and only observable through noisy sensors.

5.1 SMAC with State Uncertainty

MessySMAC is a modified version of SMAC with *observation stochasticity* w.r.t. Ω, where the observation values of $z_{t,i}$ are negated with a probability of $\phi \in [0, 1)$, and *initialization stochasticity* w.r.t. b_0, where K random steps are initially performed before officially starting an episode. MessySMAC represents a more general Dec-POMDP challenge which enables systematic evaluation under various state uncertainty configurations according to ϕ and K.

Fig. 2 shows the PCA visualization of joint observations in two maps of SMAC ($K = 0$) and MessySMAC ($K = 10$) within the first 5 steps of 1,000 episodes using a random policy. While the observations of the initial state s_0 (dark purple) in original SMAC are very similar and can be easily distinguished from subsequent steps, the separability in MessySMAC is much harder due to significantly higher entropy in b_0, indicating higher initialization stochasticity.

5.2 Comparison with SMACv2

SMACv2 is an update to the original SMAC benchmark featuring initialization stochasticity w.r.t. position and unit types, as well as observation restrictions [6]. SMACv2 addresses similar issues as MessySMAC but MessySMAC additionally features *observation stochasticity* w.r.t. Ω according to the general Dec-POMDP formulation in Section 2.1. Unlike MessySMAC, SMACv2 does not support the *original SMAC maps* therefore not enabling direct comparability w.r.t. various state uncertainty configurations.

Thus, SMACv2 can be viewed as entirely new StarCraft II benchmark, while MessySMAC represents a *SMAC extension*, enabling systematic evaluation under various state uncertainty configurations for the original SMAC maps.

6 EXPERIMENTS

We use the state-based CTDE implementations of `QPLEX`, `CW-QMIX`, `OW-QMIX`, and `QMIX` from [24] as state-of-the-art baselines with the default hyperparameters. We also integrate `MAPPO` from [36].

`AERIAL` is implemented[3] using `QMIX` as factorization operator Ψ according to Fig. 1. We also experimented with `QPLEX` as alternative with no significant difference in performance. Thus, we stick with `QMIX` for efficiency due to fewer trainable parameters. The transformer of `AERIAL` has 4 heads with W_q^c, W_k^c, and W_v^c each having one hidden layer of $d_{att} = 64$ units with ReLU activation. The subsequent MLP layers have 64 units with ReLU activation.

For ablation study, we implement `AERIAL (no attention)`, which trains Ψ directly on $\mathbf{h_t}$ without self-attention as described in Section 4.2, and `AERIAL (raw history)`, which trains Ψ on the raw joint history $\boldsymbol{\tau_t}$ concatenated with the true state s_t as originally proposed for actor-critic methods [18].

6.1 Dec-Tiger

Setting. We use the Dec-Tiger problem described in Section 4.1 and [20] as simple proof-of-concept domain with $T = 4$ and $\gamma = 1$. We also provide the optimal value of 4.8 computed with MAA* for comparison [29].

[3] Code is available at https://github.com/thomyphan/messy_smac. Further experimental details are in appendix C.

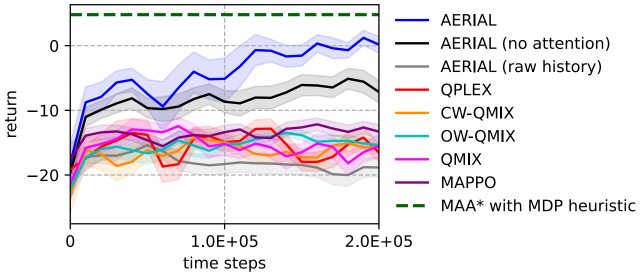


Figure 3: Average learning progress w.r.t. the return of AERIAL variants and state-of-the-art baselines in Dec-Tiger over 50 runs. Shaded areas show the 95% confidence interval.

Results. The results are shown in Fig. 3. AERIAL comes closest to the optimum, achieving an average return of about zero. AERIAL (no attention) performs second best with an average return of about -8, while all other approaches achieve an average return of about -15.

Discussion. The results confirm the example from Section 4.1 and the findings of [18, 22]. All state-based CTDE approaches and AERIAL (raw history) converge to a one-step policy, where both agents optimistically open the same door regardless of any observation. AERIAL (no attention) converges to a local optimum most of the time, where both agents only listen for all $T = 4$ time steps. AERIAL performs best due to considering the latent dependencies of all memory representations $h_{t,i} \in \mathbf{h_t}$ via self-attention to learn an adequate representation of recurrence $\mathbf{P}^{\pi}(\boldsymbol{\tau}_\mathbf{t}|b_0)$ according to Eq. 5.

6.2 Original SMAC

Setting. To directly assess the competitiveness of AERIAL against the state-of-the-art baselines, we evaluate in the original SMAC maps 3s5z and 10m_vs_11m which are classified as *easy*, as well as the *hard* maps 2c_vs_64zg, 3s_vs_5z, and 5m_vs_6m, and the *super hard* map 3s5z_vs_3s6z [26].

Results. The average test win rates at the end of training for each SMAC map are shown in Table 1. AERIAL and AERIAL (no attention) achieve competitive performance compared to QPLEX and QMIX in the easy maps 3s5z and 10m_vs_11m, while performing best in 3s_vs_5z and 5m_vs_6m. In the *super hard* scenario, AERIAL, AERIAL (no attention), and MAPPO are the only approaches achieving an average test win rate of more than 15%. However in the hard map 2c_vs_64zg, QMIX, OW-QMIX, and MAPPO outperform AERIAL and AERIAL (no attention). AERIAL (raw history) performs worst in most maps.

Discussion. The results in Table 1 show that AERIAL and AERIAL (no attention) are able to compete with state-of-the-art baselines in the original SMAC benchmark without sacrificing performance when replacing the true state s_t with the joint memory representation $\mathbf{h_t}$ in CTDE. Despite being able to outperform most baselines in the hard and super hard maps except 2c_vs_64zg, we do not claim significant outperformance, since we regard most SMAC maps as widely solved by the community [6]. AERIAL (raw history) is unable to find any meaningful policy, possibly due to the high dimensionality of $\boldsymbol{\tau}_\mathbf{t}$ and s_t which are harder to process than the more compact yet informative representations of memory $h_{t,i} \in \mathbf{h_t}$ and recurrence $\mathbf{P}^{\pi}(\boldsymbol{\tau}_\mathbf{t}|b_0)$.

6.3 MessySMAC

Setting. Analogously to SMAC, we evaluate AERIAL in the MessySMAC maps 3s5z, 10m_vs_11m, 2c_vs_64zg, 3s_vs_5z, 5m_vs_6m, and 3s5z_vs_3s6z. We set $\phi = 15\%$ and $K = 10$.

Results. The results for MessySMAC are shown in Fig. 4. AERIAL performs best in all maps with AERIAL (no attention) being second best except in 2c_vs_64zg. In the symmetric map 3s5z and asymmetric map 3s_vs_5z, AERIAL (no attention) performs almost as well as AERIAL. QMIX and QPLEX are the best performing state-of-the-art baselines in most maps. In the super hard map 3s5z_vs_3s6z, only AERIAL and AERIAL (no attention) are able to progress notably. AERIAL (raw history) performs worst in all maps. MAPPO only progresses notably in 2c_vs_64zg.

Discussion. Similar to the Dec-Tiger experiment, the results confirm the benefit of exploiting more accurate closed-loop information like memory representations in domains with high state uncertainty. AERIAL consistently outperforms AERIAL (no attention), indicating that self-attention can correct for the naive independence assumption of all $h_{t,i} \in \mathbf{h_t}$. MAPPO performs especially poorly in MessySMAC due to its misleading dependence on true states without any credit assignment, confirming the findings of [6].

6.4 State Uncertainty Robustness

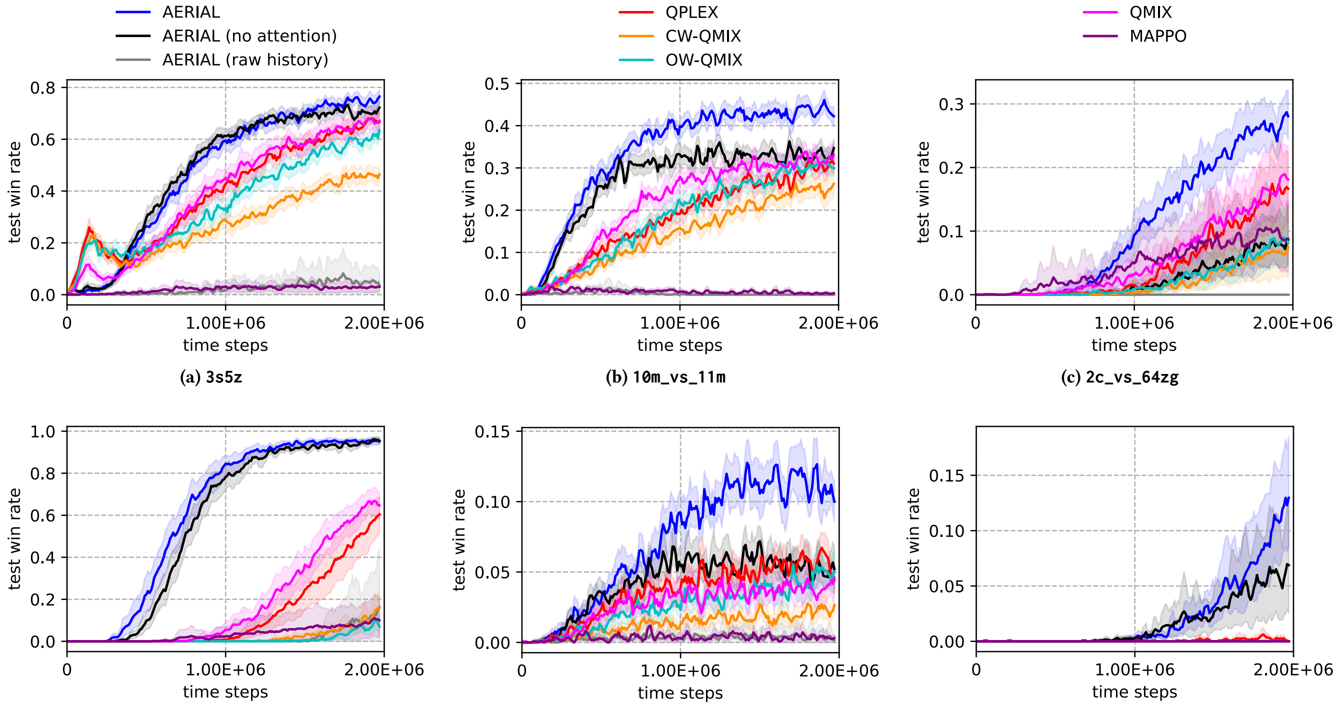
Setting. To evaluate the robustness of AERIAL and AERIAL (no attention) against various configurations of state uncertainty in MessySMAC, we manipulate Ω through the observation negation probability ϕ and b_0 through the number of initial random steps K as defined in Section 5.1. We compare the results with QMIX and QPLEX as the best performing state-of-the-art baselines in MessySMAC according to the results in Section 6.3. We present summarized plots, where the results are aggregated accross all maps from Section 6.3. To avoid that easy maps dominate the average win rate due to all approaches achieving high values there, we normalize all win rates by the maximum win rate achieved in the respective map for all tested configurations of ϕ and K. Thus, we ensure an equal weighting regardless of the particular difficulty level. If not mentioned otherwise, we set $\phi = 15\%$ and $K = 10$ as default parameters based on Section 6.3.

Results. The results regarding observation stochasticity w.r.t. Ω and ϕ are shown in Fig. 5. Fig. 5a shows that the average win rates of all approaches decrease with increasing ϕ with AERIAL consistently achieving the highest average win rate in all configurations. Fig. 5b shows that AERIAL performs best in most MessySMAC maps, especially when $\phi \geq 15\%$. AERIAL (no attention) performs second best.

The results regarding initialization stochasticity w.r.t. b_0 and K are shown in Fig. 6. Analogously to Fig. 5, Fig. 6a shows that the average (normalized) win rates of all approaches decrease with increasing K with AERIAL consistently achieving the highest average win rate in all configurations. Fig. 6b shows that AERIAL performs

Table 1: Average win rate of AERIAL variants and state-of-the-art baselines after 2 million time steps of training across 400 final test episodes for the original SMAC maps with the 95% confidence interval. The best results per map are highlighted in boldface and blue.

	AERIAL variants			state-of-the-art baselines				
	AERIAL	no attention	raw history	QPLEX	CW-QMIX	OW-QMIX	QMIX	MAPPO
3s5z	**0.95 ± 0.01**	0.90 ± 0.03	0.18 ± 0.11	0.94 ± 0.01	0.87 ± 0.02	0.91 ± 0.02	**0.95 ± 0.01**	68.7 ± 0.94
10m_vs_11m	**0.97 ± 0.01**	0.88 ± 0.04	0.09 ± 0.14	0.90 ± 0.02	0.91 ± 0.02	0.96 ± 0.01	0.90 ± 0.02	77.3 ± 0.66
2c_vs_64zg	0.52 ± 0.11	0.29 ± 0.14	0.02 ± 0.03	0.29 ± 0.1	0.38 ± 0.12	0.55 ± 0.13	0.59 ± 0.11	**90.2 ± 0.24**
3s_vs_5z	**0.96 ± 0.02**	**0.96 ± 0.02**	0.38 ± 0.13	0.74 ± 0.11	0.18 ± 0.06	0.08 ± 0.04	0.81 ± 0.05	73.8 ± 0.44
5m_vs_6m	**0.77 ± 0.03**	0.71 ± 0.04	0.1 ± 0.11	0.66 ± 0.04	0.41 ± 0.04	0.55 ± 0.06	0.67 ± 0.05	60.6 ± 1.13
3s5z_vs_3s6z	0.18 ± 0.09	**0.21 ± 0.15**	0.0 ± 0.0	0.1 ± 0.03	0.0 ± 0.0	0.02 ± 0.01	0.02 ± 0.02	20.5 ± 2.91



(a) 3s5z

(b) 10m_vs_11m

(c) 2c_vs_64zg

(d) 3s_vs_5z

(e) 5m_vs_6m

(f) 3s5z_vs_3s6z

Figure 4: Average learning progress w.r.t. the win rate of AERIAL variants and state-of-the-art baselines in MessySMAC for 2 million steps over 20 runs. Shaded areas show the 95% confidence interval. The legend at the top applies across all plots.

best in most MessySMAC maps, especially when $K \geq 10$. AERIAL (no attention) performs second best.

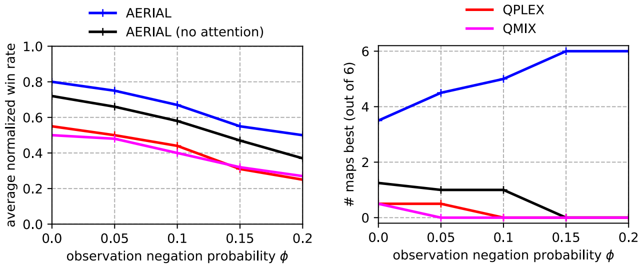
Discussion. Our results systematically demonstrate the robustness of AERIAL and AERIAL (no attention) against various configurations of state uncertainty according to Ω and b_0. State-based CTDE is notably less effective in settings, where observation and initialization stochasticity is high. As AERIAL consistently performs best in all maps when $\phi \geq 15\%$ or $K \geq 10$, we conclude that providing an adequate representation of $\mathbf{P}^{\pi}(\boldsymbol{\tau}_t|b_0)$ according to Eq. 5 learned through $\mathbf{h}_t$ and self-attention is more beneficial for CTDE than merely relying on true states when facing domains with high degrees of state uncertainty.

7 CONCLUSION AND FUTURE WORK

To tackle multi-agent problems that are messy and only observable through noisy sensors, we need adequate algorithms and benchmarks considering state uncertainty.

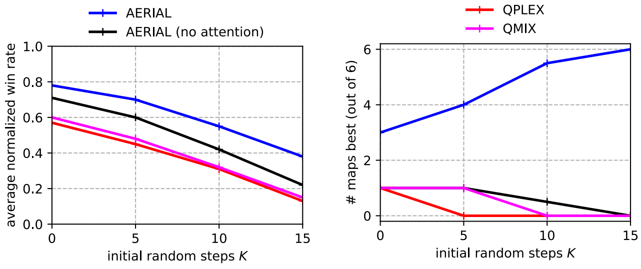
In this paper, we proposed AERIAL to approximate value functions with a learned representation of multi-agent recurrence, considering more accurate closed-loop information about decentralized agent decisions than state-based CTDE.

We then introduced *MessySMAC*, a modified version of SMAC with stochastic observations and higher variance in initial states, to provide a more general and configurable Dec-POMDP benchmark regarding state uncertainty. We showed visually in Fig. 2 and experimentally in Section 6 that MessySMAC scenarios pose



(a) average normalized test win rate **(b) # maps best out of 6 scenarios**

Figure 5: Evaluation of `AERIAL`, `AERIAL (no attention)`, and the best MessySMAC baselines for different observation negation probabilities ϕ affecting observation stochasticity w.r.t. Ω (20 runs per configuration). (a) The average normalized test win rate accross all 6 MessySMAC maps from Section 6.3. (b) The number of maps best out of 6. The legend at the top applies across all plots.



(a) average normalized test win rate **(b) # maps best out of 6 scenarios**

Figure 6: Evaluation of `AERIAL`, `AERIAL (no attention)`, and the best MessySMAC baselines for different initial random steps K affecting initialization stochasticity w.r.t. b_0 (20 runs per configuration). (a) The average normalized test win rate accross all 6 MessySMAC maps from Section 6.3. (b) The number of maps best out of 6. The legend at the top applies across all plots.

a greater challenge than their original SMAC counterparts due to observation and initialization stochasticity.

Compared to state-based CTDE, AERIAL offers a simple but effective approach to general Dec-POMDPs, being competitive in original SMAC and superior in Dec-Tiger and MessySMAC, which both exhibit observation and initialization stochasticity in contrast to original SMAC. Simply replacing the true state with memory representations can already improve performance in most scenarios, confirming the need for more accurate closed-loop information about decentralized agent decisions. Self-attention can correct for the naive independence assumption of agent-wise recurrence to further improve performance, especially when observation or initialization stochasticity is high.

We plan to further evaluate AERIAL in SMACv2 and mixed competitive-cooperative settings [17].

REFERENCES

[1] Christopher Amato, Daniel S Bernstein, and Shlomo Zilberstein. 2007. Optimizing Memory-Bounded Controllers for Decentralized POMDPs. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*. 1–8.

[2] Daniel S Bernstein, Eric A Hansen, and Shlomo Zilberstein. 2005. Bounded Policy Iteration for Decentralized POMDPs. In *IJCAI*. 52–57.

[3] Craig Boutilier. 1996. Planning, Learning and Coordination in Multiagent Decision Processes. In *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*. Morgan Kaufmann Publishers Inc., 195–210.

[4] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision Transformer: Reinforcement Learning via Sequence Modeling. In *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (Eds.), Vol. 34. Curran Associates, Inc., 15084–15097.

[5] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. 103–111.

[6] Benjamin Ellis, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob N. Foerster, and Shimon Whiteson. 2022. SMACv2: An Improved Benchmark for Cooperative Multi-Agent Reinforcement Learning. https://arxiv.org/abs/2212.07489

[7] Rosemary Emery-Montemerlo, Geoff Gordon, Jeff Schneider, and Sebastian Thrun. 2004. Approximate Solutions for Partially Observable Stochastic Games with Common Payoffs. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1* (New York, New York) *(AAMAS '04)*. IEEE Computer Society, USA, 136–143.

[8] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual Multi-Agent Policy Gradients. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018).

[9] Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative Multi-Agent Control using Deep Reinforcement Learning. In *Autonomous Agents and Multiagent Systems*. Springer, 66–83.

[10] Matthew Hausknecht and Peter Stone. 2015. Deep Recurrent Q-Learning for Partially Observable MDPs. In *2015 AAAI Fall Symposium Series*.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[12] Hengyuan Hu and Jakob N Foerster. 2019. Simplified Action Decoder for Deep Multi-Agent Reinforcement Learning. In *ICLR 2019*.

[13] Shariq Iqbal, Christian A Schroeder De Witt, Bei Peng, Wendelin Boehmer, Shimon Whiteson, and Fei Sha. 2021. Randomized Entity-wise Factorization for Multi-Agent Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 4596–4606.

[14] Shariq Iqbal and Fei Sha. 2019. Actor-Attention-Critic for Multi-Agent Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 2961–2970.

[15] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artificial intelligence* 101, 1-2 (1998), 99–134.

[16] Muhammad Junaid Khan, Syed Hammad Ahmed, and Gita Sukthankar. 2022. Transformer-Based Value Function Decomposition for Cooperative Multi-Agent Reinforcement Learning in StarCraft. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 18, 1 (Oct. 2022), 113–119.

[17] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc.

[18] Xueguang Lyu, Andrea Baisero, Yuchen Xiao, and Christopher Amato. 2022. A Deeper Understanding of State-Based Critics in Multi-Agent Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 9 (Jun. 2022), 9396–9404. https://doi.org/10.1609/aaai.v36i9.21171

[19] Xueguang Lyu, Yuchen Xiao, Brett Daley, and Christopher Amato. 2021. Contrasting Centralized and Decentralized Critics in Multi-Agent Reinforcement Learning. In *Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems*. 844–852.

[20] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. 2003. Taming Decentralized POMDPs: Towards Efficient Policy Computation for Multiagent Settings. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (Acapulco, Mexico) *(IJCAI'03)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 705–711.

[21] Frans A Oliehoek and Christopher Amato. 2016. *A Concise Introduction to Decentralized POMDPs*. Vol. 1. Springer.

[22] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. 2008. Optimal and Approximate Q-Value Functions for Decentralized POMDPs. *Journal of Artificial Intelligence Research* 32 (2008), 289–353.

[23] Thomy Phan, Fabian Ritz, Lenz Belzner, Philipp Altmann, Thomas Gabor, and Claudia Linnhoff-Popien. 2021. VAST: Value Function Factorization with Variable Agent Sub-Teams. In *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (Eds.), Vol. 34. Curran Associates, Inc., 24018–24032.

[24] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted QMIX: Expanding Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 10199–10210.

[25] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 4295–4304.

[26] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems* (Montreal QC, Canada) *(AAMAS '19)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2186–2188.

[27] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 5887–5896.

[28] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 2018. Value-Decomposition Networks for Cooperative Multi-Agent Learning based on Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems* (Stockholm, Sweden) *(AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2085–2087.

[29] Daniel Szer, François Charpillet, and Shlomo Zilberstein. 2005. MAA*: A Heuristic Search Algorithm for Solving Decentralized POMDPs *(UAI'05)*. AUAI Press, Arlington, Virginia, USA, 576–583.

[30] Ming Tan. 1993. Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning* (Amherst, MA, USA) *(ICML'93)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 330–337.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc.

[32] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. 2019. Grandmaster Level in StarCraft II using Multi-Agent Reinforcement Learning. *Nature* (2019), 1–5.

[33] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. 2021. QPLEX: Duplex Dueling Multi-Agent Q-Learning. In *International Conference on Learning Representations*.

[34] Christopher JCH Watkins and Peter Dayan. 1992. Q-Learning. *Machine Learning* 8, 3-4 (1992), 279–292.

[35] Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. 2022. Multi-Agent Reinforcement Learning is a Sequence Modeling Problem. *arXiv preprint arXiv:2205.14953* (2022).

[36] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In *36th Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A DEC-TIGER EXAMPLE

Given the Dec-Tiger example from Section 4.1 with a horizon of $T = 2$, the tiger being behind the right door (s_R), and both agents having listened in the first step, where agent 1 heard z_L and agent 2 heard z_R: The final state-based values are defined by $Q^*_{MDP}(s_t, \mathbf{a_t}) = \mathcal{R}(s_t, \mathbf{a_t})$.

Due to both agents perceiving different observations, i.e., z_L and z_R respectively, the probability of being in state s_R is 50% according to the belief state, i.e., $b(s_R|\boldsymbol{\tau}_\mathbf{t}) = b(s_L|\boldsymbol{\tau}_\mathbf{t}) = \frac{1}{2}$. Thus, the true optimal Dec-POMDP values for the final time step are defined by:

$$\begin{aligned} Q^*(\boldsymbol{\tau}_\mathbf{t}, \mathbf{a_t}) &= \sum_{s_t \in \mathcal{S}} b(s_t|\boldsymbol{\tau}_\mathbf{t})\mathcal{R}(s_t, \mathbf{a_t}) \\ &= \frac{1}{2}(Q^*_{MDP}(s_L, \mathbf{a_t}) + Q^*_{MDP}(s_R, \mathbf{a_t})) \end{aligned} \tag{10}$$

The values of Q^*_{MDP} and Q^* for the final time step $t = 2$ in the example are given in Table 2. Both agents can reduce the expected penalty when always performing the same action. Therefore, it is likely for MARL to converge to a joint policy that recommends the same actions for both agents, especially when synchronization techniques like parameter sharing are used [9, 30, 36].

Table 2: The values of Q^*_{MDP} and Q^* for the final time step $t = 2$ in the Dec-Tiger example from Section 4.1.

$\mathbf{a_t}$	$Q^*_{MDP}(s_L, \mathbf{a_t})$	$Q^*_{MDP}(s_R, \mathbf{a_t})$	$Q^*(\boldsymbol{\tau}_\mathbf{t}, \mathbf{a_t})$
$\langle li, li \rangle$	**−2**	**−2**	**−2**
$\langle li, o_L \rangle$	-101	+9	-46
$\langle li, o_R \rangle$	+9	-101	-46
$\langle o_L, li \rangle$	-101	+9	-46
$\langle \mathbf{o_L}, \mathbf{o_L} \rangle$	**−50**	**+20**	**−15**
$\langle o_L, o_R \rangle$	-100	-100	-100
$\langle o_R, li \rangle$	+9	-101	-46
$\langle o_R, o_L \rangle$	-100	-100	-100
$\langle \mathbf{o_R}, \mathbf{o_R} \rangle$	**+20**	**−50**	**−15**

B FULL ALGORITHM OF AERIAL

The complete formulation of AERIAL is given in Algorithm 1. Note that AERIAL does not depend on true states s_t at all, since the experience samples e_t (Line 20) used for training do not record any states.

C EXPERIMENT DETAILS

C.1 Computing infrastructure

All training and test runs were performed in parallel on a computing cluster of fifteen x86_64 GNU/Linux (Ubuntu 18.04.5 LTS) machines with i7-8700 @ 3.2GHz CPU (8 cores) and 64 GB RAM. We did not use any GPU in our experiments.

C.2 Hyperparameters and Neural Network Architectures

Our experiments are based on PyMARL and the code from [24] under the Apache License 2.0. We use the default setting from the paper without further hyperparameter tuning as well as the same neural network architectures for the agent RNNs, i.e., *gated recurrent units (GRU)* of [5] with 64 units, and the respective factorization operators Ψ as specified by default for each state-of-the-art baseline in Section 6. We set the loss weight $\alpha = 0.75$ for CW-QMIX and OW-QMIX.

For MAPPO, we use the hyperparameters suggested in [36] for SMAC, where we set the clipping parameter to 0.1 and use an epoch count of 5. The parameter λ for generalized advantage estimation is set to 1. The centralized critic has two hidden layers of 128 units with ReLU activation, a single linear output, and conditions on *agent-specific global states* which concatenate the global state and the individual observation per agent. The policy network of MAPPO has a similar recurrent architecture like the local utility functions Q_i and additionally applies softmax to the output layer.

AERIAL is implemented using QMIX as factorization operator Ψ according to Fig. 1. We also experimented with QPLEX as alternative with no significant difference in performance. Thus, we stick with QMIX for computational efficiency due to fewer trainable parameters. The transformer has $C = 4$ heads $c \in \{1, ..., C\}$ with respective MLPs W^c_q, W^c_k, and W^c_v, each having one hidden layer of $d_{att} = 64$ units with ReLU activation. The three subsequent MLP layers of Line 19 in Algorithm 1 have 64 units with ReLU activation.

All neural networks are trained using RMSProp with a learning rate of 0.0005.

Algorithm 1 AERIAL

```
Initialize parameters for $\langle Q_i \rangle_{i \in \mathcal{D}}$ and $\Psi$.
for episode $m \leftarrow 1, E$ do
    Sample $s_0$, $\mathbf{z_0}$, and $\boldsymbol{\tau_0}$ via $b_0$ and $\Omega$
    for time step $t \leftarrow 0, T-1$ do
        for agent $i \in \mathcal{D}$ do
            $a_{t,i} \leftarrow \pi_i(\tau_{t,i})$                    ▷ $argmax_{a_{t,i} \in \mathcal{A}_i} Q_i(\tau_{t,i}, a_{t,i})$
            $rand \sim U(0,1)$                    ▷ Sample from uniform distribution
            if $rand \leq \epsilon$ then                    ▷ $\epsilon$-greedy exploration
                Select random action $a_{t,i} \in \mathcal{A}_i$
        $\mathbf{a_t} \leftarrow \langle a_{t,i} \rangle_{i \in \mathcal{D}}$
        Execute joint action $\mathbf{a_t}$
        $s_{t+1} \sim \mathcal{T}(s_{t+1} | s_t, \mathbf{a_t})$
        $\mathbf{z_{t+1}} \sim \Omega(\mathbf{z_{t+1}} | \mathbf{a_t}, s_{t+1})$
        $\mathbf{h_t} \leftarrow \langle h_{t,i} \rangle_{i \in \mathcal{D}}$                    ▷ Query all memory representations
        Detach $\mathbf{h_t}$ from computation graph
        $\boldsymbol{\tau_{t+1}} \leftarrow \langle \boldsymbol{\tau_t}, \mathbf{a_t}, \mathbf{z_{t+1}} \rangle$                    ▷ Concatenate $\boldsymbol{\tau_t}$, $\mathbf{a_t}$, and $\mathbf{z_{t+1}}$
        for attention head $c \leftarrow 1, C$ do                    ▷ Process multi-agent recurrency
            $attention_c \leftarrow att_c(\mathbf{h_t})$                    ▷ See Eq. 9
        $rec_t \leftarrow MLP(\sum_{c=1}^{C} attention_c)$                    ▷ See Section 4.2
        $e_t \leftarrow \langle \boldsymbol{\tau_t}, \mathbf{a_t}, r_t, \mathbf{z_{t+1}}, rec_t \rangle$
        Store experience sample $e_t$
    Train $\Psi$ and $\langle Q_i \rangle_{i \in \mathcal{D}}$ using all $e_t$                    ▷ See Fig. 1
```