Consistent epistemic planning for MADRL

Peiliang Wu (✉ peiliangwu@ysu.edu.cn)  
Yanshan University

ShiCheng Luo (✉ 2522079421@qq.com)  
Yanshan University

LiQiang Tian (✉ 1090558086@qq.com)  
Yanshan University

BingYi Mao (✉ ysdxmby@163.com)  
Yanshan University

enbai Chen (✉ chenwb@bistu.edu.cn)  
Beijing Information Science & Technology University

Research Article

Keywords: Multi-agent Deep Reinforcement Learning, Multi-agent Epistemic Planning, Shared Mental Model, SMM-MEPP

DOI: https://doi.org/10.21203/rs.3.rs-2576428/v2

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License

Additional Declarations: No competing interests reported.
Consistent Epistemic Planning for Multiagent Deep Reinforcement Learning

Peiliang Wu\textsuperscript{1,2}, ShiCheng Luo\textsuperscript{1,2*†}, LiQiang Tian\textsuperscript{1,2†}, BingYi Mao\textsuperscript{1,2†} and Wenbai Chen\textsuperscript{3†}

\textsuperscript{1}School of Information Science and Engineering, Yanshan University, Qinhuangdao, 066004, Hebei, China.
\textsuperscript{2}The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Qinhuangdao, 066004 Hebei, China.
\textsuperscript{3}School of Automation, Beijing Information Science and Technology University, 100192, Beijing, China.

*Corresponding author(s). E-mail(s): 2522079421@qq.com; Contributing authors: peiliangwu@ysu.edu.cn; 1090558086@qq.com; ysdxmb@163.com; chenwb@bistu.edu.cn;
\textsuperscript{†}These authors contributed equally to this work.

Abstract

Multiagent cooperation in a partially observable environment without communication is difficult because of the uncertainty of agents. Traditional multiagent deep reinforcement learning (MADRL) algorithms fail to address this uncertainty. We proposed a MADRL-based policy network architecture called shared mental model C multiagent epistemic planning policy (SMM-MEPP) to resolve this issue. First, this architecture combines multiagent epistemic planning and MADRL to create a “perception–planning–action” multiagent epistemic planning framework, helping multiple agents better handle uncertainty in the absence of coordination. Additionally, by introducing mental models and describing them as neural networks, the parameter-sharing mechanism is used to create shared mental models, maintain the consistency of multiagent planning under the condition of no communication, and improve the efficiency of cooperation. Finally, we applied the SMM-MEPP architecture to three advanced MADRL algorithms and conducted comparative experiments in multiagent cooperation tasks.
The results show that the proposed method can provide consistent planning for multiple agents and improve the convergence speed or training effect in a partially observable environment without communication.

**Keywords:** Multiagent Deep Reinforcement Learning, Multiagent Epistemic Planning, Shared Mental Model, SMM-MEPP

1 Introduction

The group activities of humans are equivalent to a multiagent system in a partially observable environment. When people are forbidden from communicating, decisions are generally based on their cognition (or beliefs) of the observed information, and such decisions typically involve a plan (Fabiano et al., 2021). Beliefs are like human epistemic biases about things, and so are multiagents. The uncertainty problem in partially observable environments is a major challenge in multiagent deep reinforcement learning (MADRL) research (Yang et al., 2021). To address this, the traditional MADRL algorithm typically employs the framework of centralized training and decentralized execution (CTDE) (Ikeda and Shibuya, 2022) to achieve implicit communication among agents in a partially observable environment without communication. The control strategy of the agent is reflected in the policy network in such a framework, and the process of forming the policy network strategy can be simplified as **perception-action reasoning**. This entire process is unplanned, which reduces the robustness and flexibility of the agent (Muise, 2014).

Consequently, we refer to multiagent epistemic planning (MEP) (Buckingham et al., 2020) in the field of nondeep reinforcement learning and design a network structure of a multiagent epistemic planning strategy suitable for MADRL algorithm in a partially observable environment without communication: **perception-plan reasoning-action reasoning**. By reasoning about nested beliefs, MEP can generate plans that include predictive actions for multiple agents. This planning method is equivalent to segmenting the nested beliefs to better help the agent to modify the beliefs and deal with the uncertainty of the agent. In our study, an agent’s plan comprises the next task that the agent deduces and assigns to all agents. The agent then generates an action based on the corresponding task. The assignment does not send tasks to the other agents. In addition, the policy network of each agent is designed using this structure.

However, because the agents cannot communicate with each other in the environment, coordinating the plans of multiple agents is difficult. Inconsistent plans directly affect multiagent cooperation. To address this problem, we introduce a shared mental model (SMM) (Zhou, 2021), which enables multiple agents to share the same belief about the same state, resulting in multiple agents having the same cognition and deducing the same plan. We fuse the
SMM with the multiagent epistemic planning policy network structure and refer to it as the SMM-MEPP architecture.

In the actor-critic (AC) framework, the multiagent deep reinforcement learning algorithm generates actions using a policy network. Our main contribution is to design an SMM-MEPP architecture combining multiagent epistemic planning and SMMs for such a policy network and apply the SMM-MEPP architecture to three advanced AC-based MADRL algorithms. Experiments show that the SMM-MEPP architecture can help MADRL algorithms solve more difficult cooperative control tasks in a partially visible environment without requiring communication and that it is universal and effective for a variety of MADRL algorithms.

2 RELATED WORKS

The field of MADRL has rapidly developed in recent years. One of the major challenges in this field is the partially observable environment problem (Rupprecht and Wang, 2022), and one of the most common solutions is to allow information exchange between agents (Shibata et al, 2023). For instance, Foerster et al (2016) used information dissemination to compensate for an agent’s limited knowledge of the environment, although communication was achieved at high cost (Jiang and Lu, 2018). Another solution is to use memory mechanisms that are independent of communication. For example, He et al (2021) saved the relevant agent’s history information to compensate for the lack of observation data, although long-term memory may obscure past experiences and render decision-making difficult.

The multiagent epistemic planning technique addresses the environment’s partial observability by enabling a single agent to reason about the information owned by other agents (Areces et al, 2021). In the realm of non-reinforcement learning in multiagent systems, epistemic planning has yielded numerous successes. Ulusoy et al (2011) suggested a method for planning robot team movement in a public environment with temporal logic constraints. Engesser et al (2017) directly modeled potential communication behaviors as planned activities, with agents completing the plan independently, which is similar to the proposed design. In addition, a few studies in the field of MADRL used the epistemic planning approach, although the adopted methods require communication. For example, Kong et al (2017) set a method for sending planning from a high-level to a low-level agent in a multiagent deep reinforcement learning algorithm, abstracting the coordination of an agent at a low level to that at a high level.

In addition, many previous studies examined the relationship between multiagents. According to the homogeneity of multiagents, Wu et al (2021) set the policy network of all agents in the MADRL algorithm as a neural network with the same parameters. Furthermore, Iqbal and Sha (2019) did not share the homogenous agents’ policy network parameters, although it enabled agents to retain their own policy networks and learn separately. In the field of
nondeep reinforcement learning, Singh et al (2017) employed SMM method to maintain several agents with the same objective from replicating the activities of others.

In this study, we incorporated SMM into the multiagent epistemic planning framework in MADRL.

3 METHODS

The goal of our research is to create a general policy network architecture for traditional MADRL algorithms based on a partially observable multiagent environment with no communication, enabling multiagents to learn programmatic strategies in a perceptually constrained environment.

To meet the needs of future studies, some limitations of SMM-MEPP architecture are described here. Although the agent can obtain only part of the global information in a multiagent environment, the information must include part of the information of all agents simultaneously. For example, the agent is aware of only the location of the other agents, without being aware of the direction and speed of their motion. The “common information” in the following text refers to the location in the example.

3.1 Preliminaries

**Multiagent deep reinforcement learning:** We consider a multiagent extension of Markov decision process, called decentralized partially observable Markov decision process (Dec-POMDP) (Baier et al, 2021). The Dec-POMDP is represented by tuples \((\mathbb{N}, X, \{A^i\}, \{O^i\}, \mathcal{P}, \{R^i\}, \varsigma)\), where \(\mathbb{N} = \{1, \cdots, N\}\) represents the set of \(N\) interactive agents, \(X\) represents a collection of system states not observed in the global state, \(A\) represents the set of agents individual action spaces \(A^i\), \(O\) represents the set of agents observation spaces \(O^i\), \(P\) represents the transfer probability function, \(R^i\) represents the reward function related to agent \(i\), and \(\varsigma\) represents the discount coefficient.

In a multiagent system, the state transition is the state of the environment at the next moment after all agents perform the action according to the state transition function. Uppercase letters \(S\) and \(A\) represent the state random variable and action random variable of agent \(i\), respectively, whereas the lowercase letters \(s\) and \(a\) represent the observed value of the state and action of agent \(i\), respectively. The state transition function is a conditional probability density function, denoted as follows:

\[
p (s_{t+1} \mid s_t; a_t) = P[S_{t+1} = s_{t+1} \mid S_t = s_t, A_t = a_t] \quad (1)
\]

In other words, the state \(S_{t+1}\) at the next moment depends on the state \(S_t\) at the current moment and action \(A_t = [A^1_t, A^2_t, \cdots, A^m_t]\) of \(m\) agents.

Reward is the feedback provided by the environment to the agent. The reward \(R^i_t\) obtained by agent \(i\) at time \(t\) depends on the state \(S_t\) and action \(A_t\) of all the agents. We consider that the state \(s\) of a single agent is a partially
observable state $o^i$, that is, $s^i = o^i$. The policy learned by each agent is denoted as $\pi(\cdot \mid o^i; \theta^i)$, and $\theta^i$ is the parameter of the policy network of agent $i$ (deep reinforcement learning adopts a neural network to fit policy $\pi$, which is referred to as the policy network).

The goal of each agent is to learn an optimal policy network to generate optimal actions to maximize total revenue $U$.

$$U^i = \sum_{t=0}^{T} \gamma_t R^i_t$$

(2)

where $\gamma \in [0, 1]$ represents the discount rate and $T$ represents the time range.

To learn the optimal strategy network, most MADRL algorithms adopt an AC framework, combine the value network, and use the policy gradient technique to learn the parameters $\theta = [\theta^1, \theta^2, \cdots, \theta^m]$ of the strategy network. The policy gradient technique can be expressed as an optimization problem as follows:

$$\max_{\theta} \left\{ J(\theta) \triangleq E_S [V_\pi(S)] \right\}$$

(3)

where $V_\pi(S) = E_{A \sim \pi(\cdot \mid S; \theta)} [Q_\pi(S, A)]$

(4)

$Q_\pi(s, a) = E [U \mid S = s, A = a]$

(5)

where $V_\pi(S)$ represents the state value function and $Q_\pi(s, a)$ represents the action value function. The calculation of $Q_\pi(s, a)$ uses the neural network $q(s, a; \omega)$ approximation.

The optimization problem of the policy gradient technique is expressed as maximization of the expected state value by constantly adjusting the policy network parameter $\theta$. The algorithm used to solve this maximization problem is gradient ascent, which increases the objective function $J$, as follows:

$$\theta \leftarrow \theta + \beta \cdot \nabla_{\theta} J(\theta)$$

(6)

where $\beta$ represents the learning rate and $\nabla_{\theta} J(\theta)$ represents the gradient.

$$\nabla_{\theta} J(\theta) = E_S \left[ E_{A \sim \pi(\cdot \mid S; \theta)} [Q_\pi(S, A) \cdot \nabla_{\theta} \log \pi(A \mid S; \theta)] \right]$$

(7)

To calculate $\nabla_{\theta} J(\theta)$, the unbiased estimator $g(s, a; \theta)$ approximates policy gradient $\nabla_{\theta} J(\theta)$.

$$g(s, a; \theta) \triangleq Q_\pi(s, a) \cdot \nabla_{\theta} \log \pi(a \mid s; \theta)$$

(8)

**Partially observable environment:** An agent cannot observe the global environmental information or access the internal information of other agents (Gurov et al., 2022). We divide the information carried by a single agent in this environment into observable information (position coordinates) and unobservable information (dynamic information such as acceleration), and divide
the information \( o^i \) acquired by agent \( i \) into common information \( oc^i \) and private information \( op^i \), that is, \( o^i = [oc^i, op^i] \). Common information \( oc^i \) includes observable information and environmental information of all agents, whereas private information \( op^i \) includes unobservable information of agent \( i \).

### 3.2 Multiagent Epistemic Planning and Multiagent Deep Reinforcement Learning

Epistemic planning of multiagent system can also be called belief planning. Firstly, the concepts of knowledge and belief in agent cognition are introduced. When the agent observes the global state, it is said that the agent’s cognition of the global state is knowledge. When the agent can only observe the local state, it is said that the agent’s cognition of the local state is belief. In a partially observable multi-agent system, because of the uncertainty of the agent, the cognition of the observed state belongs to the belief. For example, agent \( agent_1 \) does not observe the agent \( agent_2 \), but \( agent_1 \) thinks and believes that \( agent_2 \) is in the state of \( \mathcal{E} \), no matter whether \( agent_2 \) is really in the state of \( \mathcal{E} \). A belief is a limited thought. The belief in multiagent system is nested, that is, the belief of the agent to any agent and the belief of the global environment is nested into the belief of the agent. MEP describe and reason beliefs based on dynamic epistemic logic (DEL), which enables agents to plan for multiple intelligent systems. These plans are equivalent to the segmentation of nested beliefs in the agent according to reasoning and the belief of the agent to other agents is expressed as a plan one by one. When the agent observes that its plans for an agent are inconsistent with reality, it is better able to modify local beliefs for nested beliefs.

MADRL and multiagent epistemic planning (MEP) are different solutions for multiagent systems with the goal of finding a series of optimal actions to complete the task. The difference lies in the fact that MADRL is based on Dec-POMDP and uses neural networks to reason states and information to find the optimal action, whereas MEP is based on dynamic epistemic logic (DEL) and uses logic to formally reason states and information and generate a series of plans to find the optimal action. In MEP, plans are the result of DEL reasoning, and actions are the product of plans; in MADRL, actions are the result of neural network reasoning. Therefore, although MADRL is unable to use a highly structured and formulaic DEL, DRL techniques can be used to replace DEL reasoning to generate plans, and then actual actions can be formed according to the plans. The following points are discussed from a theoretical perspective:

MEP can be represented by tuple \( \{\varrho, \hat{A}, Ag, I, \mathcal{G}\} \) (Bolander and Andersen, 2011). Here, \( \varrho, \hat{A}, and Ag \) represent the set of propositions (propositional symbols), set of action spaces, set of agents, respectively. \( I \) represents the set of initial states and \( \mathcal{G} \) represents the set of goal states. MEP is related to information changes and inherits the concept of DEL. Specifically, the cognitive language \( \mathcal{L}_{kc} \) used to represent the knowledge of agents conforms to the syntax
rules of DEL, which is defined as follows:

\[ \varphi ::= p \mid \neg \varphi \mid (\varphi \land \varphi) \mid K_{ag} \varphi \mid C\varphi \]  

(9)

where \( p \in \wp, \, ag \in Ag, \, \varphi, \phi \in \mathcal{L}_{kc} \), \( K_{ag} \varphi \) represents "agent ag knows \( \varphi \)", and \( C\varphi \) represents "\( \varphi \) is common knowledge". In the Dec-POMDP model of MADRL in Subsection 3.1, the specified \( P, \{R^i\}, \zeta \) belongs to the proposition \( \wp, \wp \) equivalent \( \wp \), \( \wp \) equivalent to \( Ag \), the initial state \( \{X, O^i\}_0 \), and the reward maximizing \( \{X, O^i\}_g \) are equivalent to \( I \) and \( G \). Therefore, MADRL is equivalent to another form of MEP; that is, MADRL is also a planning task; therefore, MADRL can theoretically use the explicit planning methods of MEP.

The MEP method used in MADRL is described as follows:

A mapping function, \( \omega : A \to Ag \) is defined to map each action to the agent that can execute it. A cooperative planning task \( \Pi = \langle s_0, A, \omega, \gamma \rangle \) is defined, where \( s_0 \) is the initial cognitive state and \( \gamma \) is the goal formula. In MEP, \( \gamma \) is determined by \( \mathcal{L}_{kc} \); therefore, in MADRL, DRL is used to replace \( \mathcal{L}_{kc} \). When \( s_0 \) is local to the agent, we refer to it as the planning task of the agent. Given a state \( s, \Pi(s) = \langle s, \wp, \omega, \gamma \rangle \), and the planning task of agent \( i \) is \( \Pi^i = \Pi(s^i) \). Given a multiagent system, each agent \( i \) must complete the planning task \( \Pi^i \) (agent \( i \) can only observe local states). The solution to the planning task \( \Pi^i \) is also called a plan and can be a sequence or policy. In this paper, this plan refers to the sequence \( (\tilde{a}_1, \ldots, \tilde{a}_n) \). In MEP, \( \tilde{a}_i \) represents the action at the \( i \)-th step, whereas in MADRL, \( \tilde{a}_i \) represents the subplan at the \( i \)-th step, and the action needs to be reasoned according to \( \tilde{a}_i, \tilde{a}_{i-1} \) satisfies the goal formula \( \gamma \) after executing the sequence \( (\tilde{a}_1, \ldots, \tilde{a}_{i-1}) \) under the initial state \( s_0 \).

\[ s_0 \otimes \tilde{a}_1 \otimes \cdots \otimes \tilde{a}_n \models \gamma \]  

(10)

### 3.3 Multiagent Epistemic Planning Policy network

MEP means that agents apply various models or methods (e.g., heuristics or DEL) according to their beliefs to infer the state of the world and the change of information, and then produce a suitable plan. In MADRL, we represent DEL and other methods as neural network models. Agents learn a network with a reasoning function through reinforcement learning, use this network to reason beliefs, and generate plans. In addition, Bolander and Andersen referred to MEP as the generation of plans for multiple agents; therefore, the proposed network model must output multiple plans.

In view of the partially observable environment without communication, we designed the three-layer structure of the multiagent epistemic planning process, *Perception–Planning–Action*, which is suitable for MADRL algorithm, and used it to reconstruct the policy network of MADRL algorithm, as shown in Figure 1.
The planning neural network in Figure 1 is used to learn the belief reasoning function, which is equivalent to heuristics and other models or methods, whereas the executive neural network is used to learn to convert plans into actual actions.

### 3.4 Multiagent epistemic planning policy network integrated with shared mental model

Although the previous section 3.3 has enabled agents to learn epistemic planning, the consistency of epistemic planning among multiple agents can only be learned slowly through constant training. Therefore, we introduced SMMs to ensure consistent planning. Mental models are simplified representations used by humans to explain or predict their surroundings (Rouse and Morris, 1986). The SMM has been shown to help team members understand and anticipate their surroundings, resulting in behaviors that are more appropriate for the task requirements (Seo et al, 2021).

Planning is a type of prediction based on its own right (Geffner and Bonet, 2013). Moreover, the purpose of the planning neural network in the network structure of the epistemic planning strategy shown in Figure 1 is to enable the agent to comprehend the state information and make pertinent predictions (or plans). Therefore, we define the mental model of agents as the planning neural network in the “perception–planning” layer. Then, Alshehri et al (2021) mentioned that the updating conditions of the SMM of multiple agents are as follows: (1) agents observe the common state, and (2) agents have common knowledge about the common state; therefore, the planning neural network only accepts common information (information that all agents know) from multiple agents, and it is renamed as the shared planning.
neural network. SMM denotes that all the agents’ shared planning neural networks have the same parameters. Each agent can produce the same plan based on the same data owing to the SMM in our architecture.

The SMM and MEPP network were combined to form the SMM-MEPP architecture. Figure 2 depicts the architecture and the main information transmission path.

According to the MADRL algorithm with the SMM-MEPP architecture, the action formation process of each agent is as follows. First, a single planner (agent) uses restricted sensors to collect local observations, both public and private. Then, the planner generates plans for numerous agents by feeding the shared data into a “shared planning neural network.” Finally, the planner extracts its own plan from all the plans and combines it with private data to feed an independent “executive neural network” to determine its next action.

3.5 Learning multiagent policy with SMM-MEPP architecture

**Fusion of MADRL algorithm and SMM-MEPP**: The MADRL algorithm that employs the CTDE paradigm enables all agents to have their own policy networks, and a policy network is composed of multiple neural network layers. The policy network structure of each agent is the same when the SMM-MEPP architecture is integrated into the MADRL algorithm and the original network is split into two networks: a shared planning neural network and an execution neural network. In addition, it is necessary to ensure that the complexity of the
neural network layer in the policy network is as similar as possible to reduce the influence of a single network layer change on the experiment and improve the influence of structuring on the experiment.

The changes before and after the framework are shown in Figure 3. We assume that three agents exist in the “simple_spread” simulation environment (see the next section for details) and the policy network structure of each agent in traditional MADRL (as shown in (a) in Figure 3). However, although each agent is aware of the coordinates of three agents and three target points (two dimensions), it is only aware of itself and does not know the dynamic information of other agents (six dimensions). Therefore, the dimension of each agent’s observation information is 18 (2 × 6 public information and 6 × 1 private information). After MADRL algorithm applies SMM-MEPP architecture, the policy network is split into two parts (as shown in Figure 3 (b)): the shared planning neural network includes parts from Linear1 to Linear3, and the executive neural network consists of Linear4 to Linear5 parts. The shared planning neural network inputs public information and outputs m plans; the executive neural network inputs private information, plans i, and outputs the action.

![Fig. 3](image)

**Fig. 3:** Changes before and after the fusion of SMM-MEPP architecture for a single agent’s policy network in MADRL algorithm

**Learning strategy:** When $\Upsilon_i = \{\Upsilon_1, \Upsilon_2, \cdots, \Upsilon_m\}$ represents the plan formed by agent $i$, then $\Upsilon = \Upsilon_1 = \Upsilon_i = \Upsilon_m (i = 1, \cdots, m)$. Agent $i$’s policy $\pi(a^i | o)$ can be expressed as

$$\pi (a^i | o) \rightarrow \pi (\cdot | op^i, oc^i) \rightarrow \pi (\cdot | op^i, \Upsilon^i)$$  \hspace{1cm} (11)

Each agent independently generates a plan. However, because all agents’ plans are identical, only one agent needs to learn the way to generate plans during
the training stage; the other agents can generate the same plans in the testing stage. The goal of policy learning is to maximize the expected value $E_S [V_{\pi} (S)]$ of the state value, as shown in Eq. 3. The state value $V_{\pi} (S^i)$ of agent $i$ can be expressed as follows:

$$V_{\pi} (S^i) = E_{A \sim \pi(\cdot | o^i, \Upsilon^i; \theta)} [Q_{\pi} (o^i, A)]$$ (12)

When $J (\theta)$ is maximized, gradient $\nabla_{\theta} J (\theta)$ is expressed as follows:

$$\nabla_{\theta} J (\theta) = E_S \left[ E_{A \sim \pi(\cdot | o^i, \Upsilon^i; \theta)} [Q_{\pi} (o^i, A) \cdot \nabla_{\theta} \log \pi (A | o^i, \Upsilon^i; \theta)] \right]$$ (13)

The executive neural network is owned by each agent, and its updating method is consistent with the original MADRL algorithm policy network according to Formula 14, where Decide denotes the executive neural network.

$$\theta^i_{\text{Decide}} \leftarrow \theta^i_{\text{Decide}} + \beta^i_{\text{Decide}} \cdot \nabla_{\theta^i_{\text{Decide}}} J (\theta)$$ (14)

In the training phase, only one agent’s planning neural network needs to be trained according to Formulas 15 and 16, in which Thought denotes the planning neural network. The plan is divided and passed to all agents; therefore, the gradient of the planning neural network is the sum of the gradients backpropagated by all agents.

$$\nabla_{\theta_{\text{Thought}}} J (\theta) = \sum_{i=1}^{m} \nabla_{\theta^i_{\text{Thought}}} J (\theta)$$ (15)

$$\theta_{\text{Thought}} \leftarrow \theta_{\text{Thought}} + \beta_{\text{Thought}} \cdot \nabla_{\theta_{\text{Thought}}} J (\theta)$$ (16)

The proposed SMM-MEPP architecture can be efficiently applied for the training of traditional MADRL algorithms. With MAAC algorithm as an example, assuming that the number of agents is $m$, the training pseudocode of SMM-MEPP+MAAC is illustrated as follows:
Algorithm 1 SMM-MEPP+MAAC

1: Initialize all network parameters
2: Initialize the state of each agent \( o_t^i = [oc_t, op_t^i] \)
3: Initialize Experience pool buffer
4: for \( b := 1 \) to MaxBatchsize do
   5:     for step:=1 to step_length do
       6:         Feed-forward: \( idea_t = Thought(oc_t) \)
       7:         for \( i := 1 \) to m do
           8:             Feed-forward: \( a_t^i = Decide(idea_t^i, op_t^i) \)
           9:         end for
       10:         Perform the action \( a_t^i \), and get the reward \( r_t^i \) and the state \( o_{t+1}^i \) at the next moment
       11:         Store experiences \( (o_t^1, a_t^1, r_t^1, o_{t+1}^2) \) to the experience pool buffer
       12:         Set gradient\(_{Thought} = 0 \)
       13:         for \( i := 1 \) to m do
           14:             Update critic network
           15:             Update Decide network (Formula 14)
           16:             Set gradient\(_{Thought}^i = \text{gradient}^{i}_{Thought} \)
           17:             Update Thought network (Formula 16)
           18:         end for
           19:     end for
       20:     Update target_critic and target_policy (including target_Thought and target_Decide).
       21: end for

4 EXPERIMENT

The experiment was conducted in a multiagent control environment, where different agents must coordinate their operations and movements to complete the task. SMM-MEPP architecture was integrated into different baseline algorithms and compared to the original algorithm in a simulation environment, as well as necessary ablation experiments (algorithms without MEP structure could not fuse SMM; therefore, ablation experiments only included MEP+ baseline algorithm without SMM).

4.1 Experimental environment

In this study, the simple_spread fully cooperative task environment in a multiagent particle environment was developed by Parnika et al (2021).
**simple_spread**: A total of $m$ cooperative agents and $m$ stationary target points exist in the environment. Multiple agents must learn to avoid other agents, quickly choose different and most appropriate targets, and move to the target point. The agent makes the next decision based only on the incomplete observational information available, and constantly adjusts its strategy according to the reward.

**Rewards**: Each agent is rewarded according to the distance between all agents and the target point. The reward is negatively correlated with distance. All agents were penalized if they collided. Each agent receives an equal share of the total reward value, which is calculated by adding the total reward value earned by each agent at a particular point in time:

**Observation**: Each agent can obtain the position of other agents, the position of $m$ target points and the dynamic information of the agent.

**Action**: Each agent’s action space is stationary, up, down, left, right.

**Uncertainty**: Agents are only aware of the position of other agents, without being aware of their current speed direction and other dynamic information, which is highly likely to lead multiple agents to the same target point, resulting in negative results such as collision.

### 4.2 Experimental details

To help readers understand and replicate our results, we provide key implementation details as follows:

**Baselines**: To test the effectiveness of SMM-MEPP on different algorithms, three of the most advanced MADRL algorithms with CTDE were selected.

**MADDP**: It is an improvement of the DDPG algorithm to adapt to the multiagent environment. The core of MADDPG is the introduction of critics, who can observe global information, to guide the training of actors, although actors still can only perform actions based on local observations. In addition, the actor network outputs only one action and each agent’s actor network remains independent (Lowe et al, 2017).
MAAC: The core of the algorithm is the introduction of the attention mechanism in critic, enabling the critic to focus more on the information from other agents when scoring the agent.

MAPPO: It is a progression of the PPO algorithm from a single agent to a multiagent environment. MAPPO uses an on-policy learning framework and an AC structure. In addition, the algorithm mentions the homogeneity of agents in the multiparticle environment; therefore, the actor network of each agent is fully shared. Although policy networks (actors) are shared, the output actions differ because each agent has different inputs (Yu et al., 2021).

Integration of SMM-MEPP architecture and baselines: All three baseline algorithms use the CTDE framework, and their integration with the SMM-MEPP architecture is implemented by modifying the policy network as described in the Figure 3. To ensure the impact of the SMM-MEPP architecture on the experimental results, the network hyperparameters (such as the number of internal parameters and linear network layers) should be kept as constant as possible before and after the algorithm fusion architecture.

4.3 Experimental results and analysis
The reward value obtained by the agent in the simulation environment reflects the comprehensive score of intelligent collision, coordination scheduling, and distance from the target. In addition, the reward curve of a single agent in multiple agents is exactly the same, because the reward of all agents in the simple_sprea d environment is shared. Therefore, this study considered the average reward value comparison curve of each episode of a single agent as the experimental result (the result graph has been smoothed). All comparison experiments were performed on the same device, and the GPU model was NVIDIA GeForce GTX 1080 Ti.

Figures 5 to 7 show different numbers of agents (3, 5, and 10, respectively) to test the effect of the SMM-MEPP architecture in different complexity environments, because the training effect of the traditional algorithm will decrease with an increase in the number of agents. In addition, we used three baseline algorithms that conformed to the CTDE framework to test the applicability of the architecture to the algorithm.

![Comparison of SMM-MEPP + MAAC, MAAC algorithm, and ablation experiment](image-url)
Consistent Epistemic Planning for Multiagent Deep Reinforcement Learning

From the experimental results shown in Figures 5 to Figure 7, the following is clear: First, SMM-MEPP architecture performs better when integrated into the three MADRL algorithms that conform with CTDE framework, whether in a simple 3-agent environment or a more complex 10-agent environment. This demonstrates that the SMM-MEPP architecture is applicable and effective for different algorithms that conform to CTDE framework and environments with different complexity levels. Second, we can observe from the ablation experiment “MEP+baselines” results that, with the exception of a few results (as shown in Figure 7, the first from the left), the results of other experiments are improved based on the baselines algorithm. This indicates that adding only epistemic planning to MADRL algorithm can improve the algorithm in a partially observable environment without communication, although it is not sufficiently stable. Furthermore, the ablation experiment shows that SMM improves the stability of the algorithm based on the MEP structure, which enables the SMM-MEPP architecture to be more applicable to MADRL algorithms that conform to the CTDE framework.

![Fig. 6: Comparison of SMM-MEPP + MADDPG, MADDPG algorithm, and ablation experiment](image1)

![Fig. 7: Comparison of SMM-MEPP + MAPPO, MAPPO algorithm, and ablation experiment](image2)

<table>
<thead>
<tr>
<th></th>
<th>3-agent</th>
<th>5-agent</th>
<th>10-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAAC</td>
<td>6%</td>
<td>13%</td>
<td>2%</td>
</tr>
<tr>
<td>MADDPG</td>
<td>13%</td>
<td>2%</td>
<td>5%</td>
</tr>
<tr>
<td>MAPPO</td>
<td>19%</td>
<td>5%</td>
<td>2%</td>
</tr>
</tbody>
</table>

Table 1: Improvement degree of SMM-MEPP to three algorithms (approximate value).
From Figures 5 to 7, we can observe that with an increase in the number of agents, the training effect of the algorithm is bound to decrease. According to Table 1, SMM-MEPP architecture does not completely decrease the degree of improvement of the algorithm. For example, the degree of improvement of the algorithm MADDPG increases from 2% for three agents to 5% for five agents. Therefore, SMM-MEPP architecture was robust to changes in the number of agents.

5 Conclusions

MAAC, MADDPG, and MAPPO algorithms provided three different solutions for partially observable environments without communication. This study proposed and successfully applied the SMM-MEPP architecture to the aforementioned three algorithms, which adds a consistent multiagent epistemic planning capability to the algorithms. When multiple agents are unable to communicate, the architecture enables them to use the SMM to provide them with the same planning capability to speed up belief correction and better handle uncertainty. Traditional MADRL usually require a large number of experiments to harmonize multiple agents to deal with uncertainty. The condition of multi-agent coordination is that the belief becomes correct. Therefore, the research of this study provides a new method to accelerate multiagent coordination for MADRL. Experiments show that the SMM-MEPP architecture is effective in certain MADRL algorithms and environments.

However, the SMM-MEPP architecture can only be applied to a multiagent cooperation task and does not involve an adversarial task. Therefore, future work will consider analyzing the ways to apply this architecture to different types of tasks.

Declarations

- Funding
  This work was Supported by the National Key R&D Program of China (2018YFB1308300), National Natural Science Foundation of China (62276028, U20A20167), Beijing Natural Science Foundation (4202026), Natural Science Foundation of Hebei Province (F202103079) and the Innovation Capability Improvement Plan Project of Hebei Province (22567626H).

- Conflict of interest/Competing interests
  The authors declare that they do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

- Ethics approval
  Not applicable

- Consent to participate
  Not applicable

- Consent for publication
  Not applicable
• Availability of data and materials
  Not applicable
• Code availability
  Not applicable
• Authors’ contributions
  Not applicable

References


Muise C (2014) Exploiting Relevance to Improve Robustness and Flexibility in Plan Generation and Execution. University of Toronto (Canada)


Rouse WB, Morris NM (1986) On looking into the black box: Prospects and limits in the search for mental models. Psychological bulletin 100(3):349


Shibata K, Jimbo T, Matsubara T (2023) Deep reinforcement learning of event-triggered communication and consensus-based control for distributed


Zhou Y (2021) Ideology, censorship, and propaganda: Unifying shared mental models. Available at SSRN 3821161