Lyapunov-Based Reinforcement Learning for Decentralized Multi-Agent Control

Qingrui Zhang[1], Hao Dong[2], and Wei Pan[3]

[1] School of Aeronautics and Astronautics, Sun Yat-Sen University, Guangzhou, China
qingrui.zhang@tudelft.nl
[2] Center on Frontiers of Computing Studies, Peking University, Beijing, China
hao.dong@pku.edu.cn
[3] Department of Cognitive Robotics, Delft University of Technology, Delft, the Netherlands
wei.pan@tudelft.nl

Abstract. Decentralized multi-agent control has broad applications, ranging from multi-robot cooperation to distributed sensor networks. In decentralized multi-agent control, systems are complex with unknown or highly uncertain dynamics, where traditional model-based control methods can hardly be applied. Compared with model-based control in control theory, deep reinforcement learning (DRL) is promising to learn the controller/policy from data without the knowing system dynamics. However, to directly apply DRL to decentralized multi-agent control is challenging, as interactions among agents make the learning environment non-stationary. More importantly, the existing multi-agent reinforcement learning (MARL) algorithms cannot ensure the closed-loop stability of a multi-agent system from a control-theoretic perspective, so the learned control polices are highly possible to generate abnormal or dangerous behaviors in real applications. Hence, without stability guarantee, the application of the existing MARL algorithms to real multi-agent systems is of great concern, e.g., UAVs, robots, and power systems, etc. In this paper, we aim to propose a new MARL algorithm for decentralized multi-agent control with a stability guarantee. The new MARL algorithm, termed as a multi-agent soft-actor critic (MASAC), is proposed under the well-known framework of "centralized-training-with-decentralized-execution". The closed-loop stability is guaranteed by the introduction of a stability constraint during the policy improvement in our MASAC algorithm. The stability constraint is designed based on Lyapunov's method in control theory. To demonstrate the effectiveness, we present a multi-agent navigation example to show the efficiency of the proposed MASAC algorithm.

Keywords: multi-agent reinforcement learning · Lyapunov stability · decentralized control · collective robotic systems

1 Introduction

Multi-agent system control has intrigued researchers from both industrial and academic communities for decades, due to its prospect in broad applications, such as formation flight of unmanned aerial vehicles (UAVs) [45, 46], coordination of multi-robots [5, 38], flocking/swarm control [35, 42], distributed sensor networks [34], large-scale power systems [18], traffic and transportation systems [7], etc. Control of a multi-agent system can be achieved in either a centralized or a decentralized manner. However, a multi-agent system with many subsystems has high state and action dimensions that will dramatically increase the design complexity and computational burdens of a single centralized controller [4]. In many applications, every agent of a multi-agent system only has local control capability with access to local observations, e.g., the cooperation of multiple vehicles [14]. The lack of global control capability and information excludes the possibility of centralized control. Besides, centralized control tends to be less reliable. If the central controller fails, the entire system will break down. As an alternative, decentralized control is capable of handling all the above issues.

Decentralized multi-agent control has been extensively studied [10, 36, 37, 43]. With the assumption that the agents' dynamics are known and linear, many model-based control algorithms have been proposed for different tasks [10, 38, 43]. In control theory, those model-based algorithms can ensure closed-loop stability if a multi-agent system satisfies all the assumptions. The state trajectories of a multi-agent system under a model-based control algorithm will always stay close to or even converge to an equilibrium point [26]. However, in most applications, agent dynamics are nonlinear, complicated, and highly uncertain, e.g., robotic systems, UAVs, and power systems. Assumptions made by model-based control algorithms can be barely satisfied in real life. Therefore, model-based control algorithms are restrictive, though theoretically sound.

Compared with model-based control, deep reinforcement learning (DRL) is more promising for the decentralized multi-agent control for complicated nonlinear dynamical systems, as it can learn controller/policy from samples without using much model information [8, 9, 13, 28, 41, 47, 48]. Recently, deep RL has obtained significant success in applying to a variety of complex single-agent control problems [3, 20, 27, 32]. However, it is more challenging to apply deep RL to decentralized multi-agent control. In multi-agent reinforcement learning (MARL), agents seek the best responses to other agents' policies. The policy update of an agent will affect the learning targets of other agents. Such interactions among agents make MARL training non-stationary, thus influencing the learning convergence. To resolve the non-stationary issue, a "centralized-training-with-decentralized-execution" mechanism was employed, based on which a number of MARL algorithms have been proposed, e.g., MADDPG [31], COMA [15], mean-field MARL [44], MATD3 [1], and MAAC [24], etc. Unfortunately, the existing MARL algorithms can not ensure the closed-loop stability for a multi-agent system, while stability is the foremost concern for the control of any dynamical systems. It is highly possible that learned control polices will generate abnormal or risky behaviors in real applications. From a control perspective, the learned control policies fail to stabilize a multi-agent system, so they cannot be applied to safety-critical scenarios, e.g., formation flight of UAVs.

In this paper, we propose MARL algorithms for decentralized multi-agent control with a stability guarantee. A multi-agent soft actor-critic (MASAC) algorithm is developed based on the well-known "centralized-training-with-decentralized-execution" scheme. The interactions among agents are characterized using graph theory [12]. Besides, a stability-related constraint is introduced to the policy improvement to ensure the closed-loop stability of the learned control policies. The stability-related constraint is designed based on the well-known Lyapunov's method in control theory which is a powerful tool for the design of a controller to stabilize the complex nonlinear systems with stability guarantee. [26].

Contributions: The contributions of this paper can be summarized as follows.

1. For the first time, a Lyapunov-based multi-agent soft actor-critic algorithm is developed for decentralized control problems based on the "centralized-training-with-decentralized-execution" to guarantee the stability of a multi-agent system.
2. Theoretical analysis is presented on the design of a stability constraint using Lyapunov's method.

2 Preliminaries

2.1 Networked Markov game

Interactions among N agents are characterized using an undirected graph $\mathcal{G} = \left\langle \mathcal{I}, \mathcal{E} \right\rangle$, where $\mathcal{I} := \{1, \ldots, N\}$ represents the set of N agents and $\mathcal{E} \subseteq \mathcal{I} \times \mathcal{I}$ denotes the interactions among agents. If an agent i is able to interact with an agent j with $j \neq i$ and i,$j \in \mathcal{I}$, there exists an edge $(i,\, j) \in \mathcal{E}$, and agent j is called a neighbor of agent i. For an undirected graph, $(j,\, i) \in \mathcal{E}$ if $(i,\, j) \in \mathcal{E}$. The neighborhood of agent i is denoted by $\mathcal{N}_i := \{\forall j \in \mathcal{I} |\, (i,\, j) \in \mathcal{E}\}$. Assume the undirected graph is fully connected, so there exists a path from each node $i \in \mathcal{I}$ to any other nodes $j \in \mathcal{I}$ [12, 17]. If an undirected graph is strongly connected, information could eventually be shared among all agents via the communication graph.

A networked Markov game with N agents is denoted by a tuple, $\mathcal{MG} := \left\langle \mathcal{G}, \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \right\rangle$, where $\mathcal{G} := \left\langle \mathcal{I}, \mathcal{E} \right\rangle$ is the communication graph among N agents, $\mathcal{S} := \bigcup_{i=1}^{N} \mathcal{S}_i$ is the entire environment space with $\mathcal{S}_i$ the local state space for agent $i \in \mathcal{I}$, $\mathcal{A} := \bigcup_{i=1}^{N} \mathcal{A}_i$ denotes the joint action space with $\mathcal{A}_i$ the local action space for agent $i \in \mathcal{I}$, $\mathcal{P} := \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ specifies the state transition probability function, and $r := \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ represents the global reward function of the entire multi-agent system. The global transition probability can, therefore, be denoted by $\mathcal{P}\left(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, \boldsymbol{a}_t\right)$. The joint action of N agents is $\boldsymbol{a} = \{a_1, \ldots, a_N\}$ where a_i denotes the action of an agent $i \in \mathcal{I}$. Accordingly, the joint policy is defined to be $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_N\}$ where π_i $(\forall i \in \mathcal{I})$ are local policies for an agent i. Hence, the global policy for the entire multi-agent system is defined to be $\pi\left(\boldsymbol{a}|\boldsymbol{s}\right) = \prod_{i=1}^{N} \pi_i\left(a_i|s_i\right)$. Assume each agent i can only obtain a local observation $s_i \in \mathcal{S}_i$ (e.g. its own states and state information of its neighbors) to make decisions at the execution.

For any given initial global state $\boldsymbol{s}_0$, the global expected discounted return following a joint policy $\boldsymbol{\pi}$ is given by

$$V\left(\boldsymbol{s}_t\right) = \sum_{t=0}^{\infty} \mathbb{E}_{\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) \sim \boldsymbol{\rho}_{\boldsymbol{\pi}}} \left[\gamma^t r\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) | \boldsymbol{s}_0\right] \tag{1}$$

where γ is a discount factor, V is the global value function and $\boldsymbol{\rho}_{\boldsymbol{\pi}}$ is the state-action marginals of the trajectory distribution induced by a global policy $\boldsymbol{\pi}$. The global action-value function (a.k.a. Q-function) of the entire system is

$$Q\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) = r\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) + \gamma \mathbb{E}_{\boldsymbol{s}_{t+1}}\left[V\left(\boldsymbol{s}_{t+1}\right)\right] \tag{2}$$

2.2 Soft actor-critic algorithm

In this paper, the soft actor-critic (SAC) algorithm will be used for the design of the multi-agent reinforcement learning algorithm. The SAC algorithm belongs to off-policy RL that is more sample efficient than on-policy RL methods [41], such as the trust region policy optimization (TRPO) [39] and the proximal policy optimization (PPO) [40]. In SAC, an expected entropy of the policy $\boldsymbol{\pi}$ is added to the value functions (1) and (2) to regulate the exploration performance at the training stage [21, 49]. The inclusion of the entropy term makes the SAC algorithm exceed both the efficiency and final performance of deep deterministic policy gradient (DDPG) [19, 29]. With the inclusion of the expected entropy, the action value function (1) to be maximized for a multi-agent system will turn into

$$V\left(\boldsymbol{s}_t\right) = \sum_{t=0}^{\infty} \mathbb{E}_{\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) \sim \boldsymbol{\rho}_{\boldsymbol{\pi}}} \left[\gamma^t \left(r\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) + \alpha \mathcal{H}\left(\boldsymbol{\pi}\left(\cdot | \boldsymbol{s}_t\right)\right)\right) | \boldsymbol{s}_0\right] \tag{3}$$

where α is the temperature parameter used to control the stochasticity of the policy by regulating the relative importance of the entropy term against the reward, and $\mathcal{H}\left(\boldsymbol{\pi}\left(\cdot | \boldsymbol{s}_t\right)\right) = -\mathbb{E}_{\boldsymbol{\pi}}\big[\log\left(\boldsymbol{\pi}\left(\cdot | \boldsymbol{s}_t\right)\right)\big]$ is the entropy of the policy $\boldsymbol{\pi}$. Accordingly, a modified Bellman backup operator is defined as

$$\mathcal{T}^{\boldsymbol{\pi}} Q\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) = r\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) + \gamma \mathbb{E}_{\boldsymbol{s}_{t+1}}\left[V\left(\boldsymbol{s}_{t+1}\right)\right] \tag{4}$$

where $V\left(\boldsymbol{s}_t\right) = \mathbb{E}_{\boldsymbol{a}_t \sim \boldsymbol{\pi}}\left[Q\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) - \alpha \log\left(\boldsymbol{\pi}\left(\boldsymbol{a}_t | \boldsymbol{s}_t\right)\right)\right]$.

2.3 Lyapunov stability in control theory

A dynamical system is called stable, if its state trajectory starting in vicinity to an equilibrium point will stay near the equilibrium point all the time. Stability is a crucial concept for the control and safety of any dynamical systems. Lyapunov stability theory provides a powerful means of stabilizing unstable dynamical systems using feedback control. The idea is to select a suitable Lyapunov function and force it to decrease along the trajectories of the system. The resulting system will eventually converge to its equilibrium. Lyapunov stability of dynamic systems at a fixed policy $\pi\left(a|s\right)$ is given by Lemma 1.

Lemma 1 (Lyapunov stability). *[26] Suppose a system is denoted by a nonlinear mapping $s_{t+1} = f(s_t, \pi(a_t|s_t))$. Let $L(s_t)$ be a continuous function such that $L(s_T) = 0$, $L(s_t) > 0$ ($\forall s_t \in \Omega$ & $s_t \neq s_T$), and $L(s_{t+1}) - L(s_t) \leq 0$ ($\forall s_t \in \Omega$) where s_T is an equilibrium state and Ω is a compact state space. Then the system $s_{t+1} = f(s_t, \pi(a_t|s_t))$ is stable around s_T and $L(s_t)$ is a Lyapunov function. Furthermore, if $L(s_{t+1}) - L(s_t) < 0$ ($\forall s_t \in \Omega$), the system is asymptotically stable around s_T.*

Note that maximizing the objective function (1) or (2) doesn't necessarily result in a policy stabilizing a dynamical system.

3 Multi-agent reinforcement learning with Lyapunov stability constraint

In this section, we will first develop a MARL algorithm based on the SAC algorithm by following a similar idea as the multi-agent deterministic policy gradient descent (MADDPG) [31]. The proposed algorithm is termed as Multi-Agent Soft Actor-Critic algorithm (MASAC). The proposed MASAC algorithm is thereafter enhanced by incorporating a carefully designed Lyapunov constraint.

3.1 Multi-agent soft actor-critic algorithm

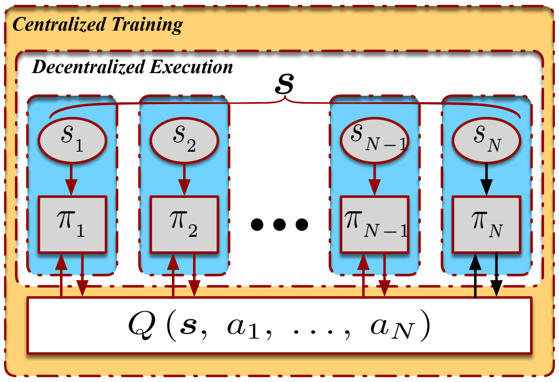

Fig. 1. Centralized training with decentralized execution

The crucial concept behind the MASAC is the so-called "*centralized training with decentralized execution*" shown in Figure 1. A centralized critic using global information is employed at the training stage, while each agent uses their own independent policy taking local observations as inputs. Hence, at the training stage, agents share their rewards with all the other agents for the calculation of the central critic. In MASAC, it is expected to maximize the entropy-regularized objective function introduced in (3).

In decentralized control, agents make decisions based on their local observations [4, 14, 25, 36, 37]. Hence, their polices are assumed to be independent of one another, i.e., $\pi\left(\boldsymbol{a}_t|\boldsymbol{s}_t\right)=\prod_{i=1}^{N}\pi_i\left(a_i|s_i\right)$. Hence, the entropy of the joint policy $\boldsymbol{\pi}\left(\boldsymbol{a}_t|\boldsymbol{s}_t\right)$ in (3) is

$$\mathcal{H}\left(\boldsymbol{\pi}\right)=-\sum_{i}^{N}\mathbb{E}_{\pi_i}\big[\log\left(\pi_i\right)\big]=\sum_{i}^{N}\mathcal{H}\left(\pi_i\right) \tag{5}$$

where $\mathcal{H}\left(\pi_i\right)$ represent the entropy of each local policy π_i.

The entire algorithm is divided into policy evaluation and policy improvement. In the policy evaluation step, we will repeatedly apply the modified Bellman backup operator (4) to the Q-value of a fixed joint policy. Let the centralized Q-value for the multi-agent system be parameterized by θ. The critic neural network parameter θ is trained to minimize the following Bellman residual.

$$\begin{aligned}J_Q\left(\theta\right)=&\mathbb{E}_{\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\sim\mathcal{D}}\left[\frac{1}{2}\Bigg(Q_\theta\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)-r\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\right.\\&\left.-\gamma\mathbb{E}_{\boldsymbol{s}_{t+1}}\left[V_{\bar{\theta}}\left(\boldsymbol{s}_{t+1}\right)+\alpha\sum_{i}^{N}\mathcal{H}\left(\pi_{\phi_i}\right)\right]\Bigg)^2\right]\end{aligned} \tag{6}$$

In the optimization, the value function is replaced by the Q-value function. Therefore, the critic parameters are optimized by stochastic gradient descent as

$$\nabla_\theta J_Q\left(\theta\right)=\mathbb{E}_{\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\sim\mathcal{D}}\Big[\nabla_\theta Q_\theta\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\delta_Q\Big] \tag{7}$$

where

$$\delta_Q=Q_\theta\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)-r-\gamma Q_{\bar{\theta}}\left(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1}\right)+\gamma\alpha\sum_{i}^{N}\log\pi_{\phi_i} \tag{8}$$

In policy improvement, the policy is updated according to

$$\boldsymbol{\pi}*=arg\min_{\boldsymbol{\pi}'\in\Pi}\mathbb{E}_{\pi_i}\Big[\alpha\sum_{i}^{N}\log\left(\pi_i\right)-Q\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\Big] \tag{9}$$

where $\boldsymbol{\pi}^*=\{\pi_1^*,\ \ldots,\ \pi_N^*\}$ is the optimal joint policy. Assume the policy of agent i is parameterized by ϕ_i, $\forall i=1,\ldots,N$. According to (9), the policy parameters ϕ_i, $\forall i=1,\ldots,N$ are trained to minimize

$$J_\pi\left(\boldsymbol{\phi}\right)\simeq\mathbb{E}_{\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\sim\mathcal{D}}\Big(\mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\phi}}}\Big(\alpha\sum_{i}^{N}\log\left(\pi_{\phi_i}\right)-Q_\theta\left(\boldsymbol{s}_t,\boldsymbol{a}_t\right)\Big)\Big) \tag{10}$$

where $\boldsymbol{\phi}=\{\phi_1,\ \ldots,\ \phi_N\}$ and $\boldsymbol{\pi}_{\boldsymbol{\phi}}=\{\pi_{\phi_1},\ \ldots,\ \pi_{\phi_N}\}$. In terms of the stochastic gradient descent, each agent's policy parameter ϕ_i will be updated according

Algorithm 1: Multi-agent soft actor-critic algorithm

```
Initialize parameters $\theta^1$, $\theta^2$ and $\phi_i$ $\forall i \in \mathcal{I}$
$\bar{\theta}^1 \leftarrow \theta^1$, $\bar{\theta}^2 \leftarrow \theta^2$, $\mathcal{D} \leftarrow \emptyset$
repeat
  for each environment step do
    $a_{i,t} \sim \pi_{\phi_i}\left(a_{i,t}|s_{i,t}\right)$, $\forall i \in \mathcal{I}$
    $\boldsymbol{s}_{t+1} \sim \mathcal{P}_i\left(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t\right)$, where $\boldsymbol{a}_t = \{a_{1,t},\ \ldots,\ a_{N,t}\}$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup \left\{\boldsymbol{s}_t, \boldsymbol{a}_t, r\left(\boldsymbol{s}_t, \boldsymbol{a}_t,\right), \boldsymbol{s}_{t+1}\right\}$
  end for
  for each gradient update step do
    Sample a batch of data, $\mathcal{B}$ , from $\mathcal{D}$
    $\theta^j \leftarrow \theta^j - \iota_Q \nabla_\theta J_Q\left(\theta^j\right)$, $j = 1, 2$
    $\phi_i \leftarrow \phi_i - \iota_\pi \nabla_{\phi_i} J_{\pi_i}\left(\phi_i\right)$, $\forall i \in \mathcal{I}$
    $\alpha \leftarrow \alpha - \iota_\alpha \nabla_\alpha J_\alpha\left(\alpha\right)$
    $\bar{\theta}^j \leftarrow \tau\theta^j + (1-\tau)\,\bar{\theta}^j$, $j = 1, 2$
  end for
until convergence
```

to

$$\nabla_{\phi_i} J_\pi\left(\boldsymbol{\phi}\right) \simeq \mathbb{E}_{\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)\sim\mathcal{D}}\Big[\Big(\nabla_{a_i} \log \pi_{\phi_i} - \nabla_{a_i} Q_{\theta_i}\left(\boldsymbol{s}_t, a_t, \bar{\boldsymbol{a}}_t\right)\Big)\nabla_{\phi_i} a_{\phi_i} + \nabla_{\phi_i} \log \pi_{\phi_i}\Big] \tag{11}$$

The temperature parameter α will be updated based on (12).

$$J_{\alpha_i} = \mathbb{E}_{\boldsymbol{\pi}}\left[-\alpha \sum_i^N \log \pi_i - \alpha\bar{\mathcal{H}}\right] \tag{12}$$

The MASAC algorithm is summarized in Algorithm 1. The final MASAC algorithm uses two soft Q-functions to mitigate the estimation bias in the policy improvement and further increase the algorithm performance [2, 16, 23].

3.2 Lyapunov stability constraint

Qualitatively, stability implies that the states of a system will be at least bounded and stay close to an equilibrium state for all the time. The existing MARL algorithms, including the proposed MASAC algorithm in Section 3.1, can find an optimal policy that can maximize either state or action-value functions. However, they do not necessarily produce a policy that ensures the stability of a system. In this section, we offer a possible solution to incorporate Lyapunov stability as a constraint in the optimization of MASAC.

A Lyapunov function candidate can be constructed based on cost functions $c\left(s, \pi\right) \geq 0$ with $c\left(s_T, \pi\left(a_T|s_T\right)\right) = 0$ and s_T the target/equilibrium state [6, 11]. One possible choice of the Lyapunov function candidate is an accumulated cost in a finite time horizon, e.g., model predictive control (MPC) [33]. Before the introduction of the Lyapunov stability constraint, we make several assumptions on both the cost functions to be designed and the system dynamics of interest. The assumption on the cost function is given as follows.

Assumption 1 *The cost function $c\left(s, \pi\left(\cdot|s\right)\right)$ is bounded $\forall s \in \Omega$ and Lipschitz continuous with respect to s, namely $\|c\left(s_1, \pi\left(\cdot|s_1\right)\right) - c\left(s_2, \pi\left(\cdot|s_2\right)\right)\|_2 \leq l_c\|s_1 - s_2\|_2$ where $l_c > 0$ is a Lipschitz constant.*

Assumption 1 could ensure the Lyapunov function candidate to be bounded and Lipschitz continuous for the state s, if we choose it to be an accumulated cost in a finite time horizon. Hence, we could further assume the Lyapunov function candidate is Lipschitz continuous with the state s on a compact set Ω with l_L the Lipchitz constant.

Since we are interested in the decentralized control problem of multiple agents with deterministic dynamics, the following assumption on the physical dynamics is made.

Assumption 2 *Consider a deterministic, discrete-time agent system $s_{t+1} = f\left(s_t, a_t\right)$. The nonlinear dynamics f is Lipschitz continuous with respect to a_t, namely $\|f\left(s_t, a_t^2\right) - f\left(s_t, a_t^1\right)\|_2 \leq l_f\|a_t^2 - a_t^1\|_2$ where $l_f > 0$ is a Lipschitz constant.*

According to the existence and uniqueness theorem, the local Lipschitz condition is a common assumption for deterministic continuous systems.

According to Lemma 1, the state s_{t+1} is needed to evaluate the stability of a system under a fixed policy, but s_{t+1} is not available in general. In Theorem 1, we show that it is possible to evaluate the stability of a new policy π_{new}, if we already have a feasible policy π_{old} associated with a Lyapunov function $L\left(s\right)$. Here, a feasible policy implies that a system is stable and that a Lyapunov function exists.

Theorem 1. *Consider a system $s_{t+1} = f\left(s_t, a_t\right)$ Suppose Assumptions 1 and 2 hold. Let π_{old} be a feasible policy for data collection and $L_{\pi_{old}}\left(s\right)$ is the Lyapunov function. A new policy π_{new} will also be a feasible policy under which the system is stable, if there exists*

$$L_{\pi_{old}}\left(s_{t+1}\right) + l_L l_f\|a_t^{\pi_{new}} - a_t^{\pi_{old}}\|_2 - L_{\pi_{old}}\left(s_t\right) \leq 0 \tag{13}$$

where $\forall s_t \in \Omega$, $\left(s_t, a_t^{\pi_{old}}, s_{t+1}\right)$ is a tuple from the policy π_{old}, l_L and l_f are Lipschitz constants of the Lyapunov function and system dynamics, respectively.

Theorem 1 requires all the states need to be visited to evaluate the stability of a new policy. Unfortunately, it is impossible to visit an infinite number of states. However, Theorem 1 still shows a potential way to use historical samples for the old policies to evaluate the current policy. Based on Theorem 1, we are able to add a Lyapunov constraint similar to (13) in the policy gradient of each agent

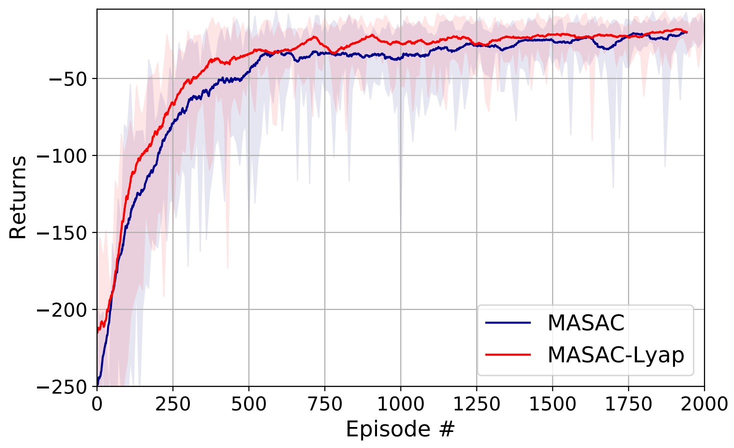

Fig. 2. Learning curves of the rendezvous experiment (40 steps per episode)

for DC of multi-agent systems. With the inclusion of the Lyapunov constraint (13) [6, 22], the objective function (10) is rewritten as

$$J_\pi(\boldsymbol{\phi}) \simeq \mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\sim\mathcal{D}}\Big(\mathbb{E}_{\boldsymbol{\pi}_\phi}\Big(\alpha\sum_i^N \log(\pi_{\phi_i}) - Q_\theta(\boldsymbol{s}_t,\boldsymbol{a}_t)\Big) + \beta\Delta L_{\boldsymbol{\phi}}\Big) \tag{14}$$

where $\beta \in [0,1]$ and $\Delta L_\psi = L_i(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1}) + l_L l_f \|\boldsymbol{a}_\phi - \boldsymbol{a}_t\|_2 - L_i(\boldsymbol{s}_t,\boldsymbol{a}_t) + \beta c_{\boldsymbol{\pi}}(\boldsymbol{s}_t)$.

At training, the Lyapunov functions $L(\boldsymbol{s}_t)$ will be parameterized by ψ which is trained to minimize

$$J_L(\psi) = \mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\sim\mathcal{D}}\left[\frac{1}{2}\left(L_\psi(\boldsymbol{s}_t,\boldsymbol{a}_t) - L_{target}\right)^2\right]$$

where $L_{target} = \sum_{t=0}^T c(\boldsymbol{s}_t,\boldsymbol{a}_t)$ with T denoting a finite time horizon as in model predictive control. The modified robust multi-agent reinforcement learning algorithm is summarized in Algorithm 2.

4 Experiment

In this section, we will evaluate our proposed algorithms in a well-known application of multi-agent systems called “rendezvous” [30]. In the “rendezvous” problem, all agents starting from different locations are required to meet at the same target location in the end. Only a subgroup of agents, which are called leaders, have access to the target location, while others need to learn to cooperate with others. In the experiments, both the critic and actor are represented suing

Algorithm 2: Multi-agent soft actor-critic algorithm with a Lyapunov constraint

```
Initialize parameters $\theta^1$, $\theta^2$ and $\phi_i$ $\forall i \in \mathcal{I}$
$\bar{\theta}^1 \leftarrow \theta^1$, $\bar{\theta}^2 \leftarrow \theta^2$, $\mathcal{D} \leftarrow \emptyset$
repeat
  for each environment step do
    $a_{i,t} \sim \pi_{\phi_i}(a_{i,t}|s_{i,t})$, $\forall i \in \mathcal{I}$
    $\boldsymbol{s}_{t+1} \sim \mathcal{P}_i(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$, where $\boldsymbol{a}_t = \{a_{1,t}, \ldots, a_{N,t}\}$
    $\mathcal{D} \leftarrow \mathcal{D} \bigcup \{\boldsymbol{s}_t, \boldsymbol{a}_t, r(\boldsymbol{s}_t, \boldsymbol{a}_t), \boldsymbol{s}_{t+1}\}$
  end for
  for each gradient update step do
    Sample a batch of data, $\mathcal{B}$ , from $\mathcal{D}$
    $\theta^j \leftarrow \theta^j - \iota_Q \nabla_{\theta_i} J_{Q_i}(\theta^j)$, $j = 1, 2$
    $\phi_i \leftarrow \phi_i - \iota_\pi \nabla_{\phi_i} J_{\boldsymbol{\pi}}(\boldsymbol{\phi})$, $\forall i \in \mathcal{I}$ from (14)
    $\alpha \leftarrow \alpha - \iota_\alpha \nabla_\alpha J_\alpha(\alpha)$
    $\bar{\theta}^j \leftarrow \tau\theta^j + (1-\tau)\bar{\theta}^j$, $j = 1, 2$
    $\psi \leftarrow \psi - \iota_L \nabla_\psi J_L(\psi)$
  end for
until convergence
```

fully connected multiple-layer perceptrons with two hidden layers. Each hidden Layer has 64 neurons with the 'ReLU' activation function. The learning rate for the actor network is chosen to be 0.0003, while the learning rate for the critic network is 0.003. To stabilize the training, learning rates decrease with a certain decay rate ($0.075^{0.0005}$ in the experiments). The Lyapunov neural network is approximated by an MLP with three hidden layers (64 neurons for the first two hidden layers, and 16 neurons for the last hidden layers). The batch size is selected to be 256. The parameter τ for soft updates of both actor and critic networks is picked to be 0.005. The discount factor γ is chosen to be 0.95.

The environment is built using the multi-agent environment used in [31]. The agent model in the environment in [31] is replaced by a high-order non-holonomic unicycle model which is widely used in robotics navigation The agent dynamics are given as follows.

$$\begin{cases} \dot{x} = v\cos\psi \\ \dot{y} = v\sin\psi \\ \dot{\psi} = \quad \omega \\ \dot{v} = \quad a \\ \dot{\omega} = \quad r \end{cases} \tag{15}$$

where x and y are the positions of the agent, ψ is the heading angle, v is the speed, and ω is the angular rate. The control actions for each agent are a and r, respectively.

Learning curves of both the MASAC and MASAC-Lyapunov are shown in Figure 2. Although both the MASAC and MASAC-Lyapunov will converge, they have a different performance at evaluations. To further verify the performance, we evaluate both the MASAC and MASAC-Lyapunov for 500 times with agents' initial positions randomly generated. We define a criterion called “success rate” to compare the overall performance of the two algorithms. We think an evaluation episode is successful if all agents end up in the target location. The “success rate” is calculated by $\frac{\text{total number of successful episodes}}{totalnumberofepisodesatevaluation} \times 100$. The “success rate” of the two algorithms is shown in Figure 3. With the inclusion of the Lyapunov constraint, we can increase the success rate of the tasks dramatically according to Figure 3. Hence, the Lyapunov constraint will increase the stability performance of the learned policy, thereby resulting in a policy that is more likely to stabilize a system.

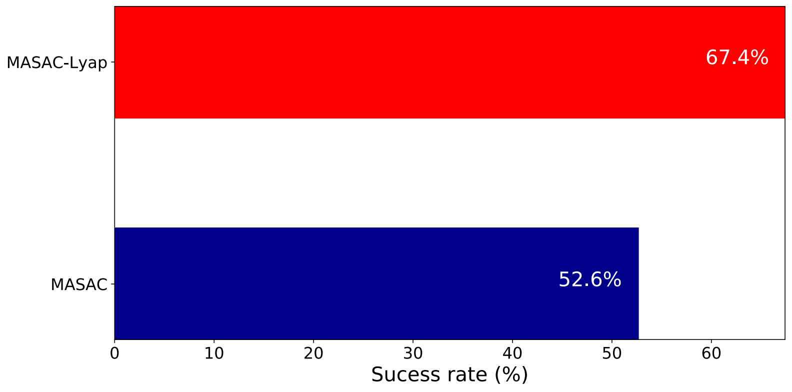

Fig. 3. Evaluation results of running the rendezvous experiment for 500 times using trained polices. (success rate $= \frac{\text{total number of successful episodes}}{\text{total number of episodes at evaluation}} \times 100$)

5 Conclusion

In this paper, we studied MARL for data-driven decentralized control for multi-agent systems. We proposed a MASAC algorithm based on the “centralized-training-with-decentralized-execution’. We thereafter presented a feasible solution to combine Lyapunov’s methods in control theory with MASAC to guarantee stability. The MASAC algorithm was modified accordingly by the introduction of a Lyapunov stability constraint. The experiment conducted in this paper demonstrated that the introduced Lyapunov stability constraint is important to design a policy to achieve better performance than our vanilla MASAC algorithm.

Bibliography

[1] Ackermann, J., Gabler, V., Osa, T., Sugiyama, M.: Reducing overestimation bias in multi-agent domains using double centralized critics. arXiv preprint arXiv:1910.01465 (2019)

[2] Ackermann, J., Gabler, V., Osa, T., Sugiyama, M.: Reducing overestimation bias in multi-agent domains using double centralized critics. ArXiv **abs/1910.01465** (2019)

[3] Andrychowicz, M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., Zaremba, W.: Learning dexterous in-hand manipulation. The International Journal of Robotics Research **30**(1), 3–20 (2020). https://doi.org/10.1177/0278364919887447

[4] Bakule, L.: Decentralized control: An overview. Annual Reviews in Control **32**, 87–98 (2008). https://doi.org/10.1016/j.arcontrol.2008.03.004

[5] van den Berg, J., Lin, M.C., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). IEEE, Pasadena, CA, USA (May 2008). https://doi.org/10.1109/ROBOT.2008.4543489

[6] Berkenkamp, F., Turchetta, M., Schoellig, A.P., Krause, A.: Safe model-based reinforcement learning with stability guarantees. arXiv preprint arXiv:1705.08551 (2017)

[7] Burmeister, B., Haddadi, A., Matylis, G.: Application of multi-agent systems in traffic and transportation. IEE Proceedings - Software Engineering **144**(1), 51–60 (Feb 1997). https://doi.org/10.1049/ip-sen:19971023

[8] Chen, Y.F., Everett, M., Liu, M., How, J.P.: Socially aware motion planning with deep reinforcement learning. In: Proceedings of 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Vancouver, BC, Canada (Sept 2017). https://doi.org/10.1109/IROS.2017.8202312

[9] Chen, Y.F., Liu, M., Everett, M., How, J.P.: Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In: Proceedings of 2017 IEEE International Conference on Robotics and Automation. Singapore, Singapore (Jun 2017). https://doi.org/10.1109/ICRA.2017.7989037

[10] Cheng, Z., Zhang, H.T., Fan, M.C., Chen, G.: Distributed consensus of multi-agent systems with input constraints: A model predictive control approach. IEEE Transactions on Circuits and Systems I: Regular Papers **62**(3), 825–834 (Mar 2015). https://doi.org/10.1109/TCSI.2014.2367575

[11] Chow, Y., Nachum, O., Duenez-Guzman, E., Ghavamzadeh, M.: A lyapunov-based approach to safe reinforcement learning. arXiv preprint arXiv:1805.07708 (2018)

[12] Diestel, R.: Graph Theory. Springer-Verlag, New York, NY, USA, 2nd edition edn. (2000)

[13] Everett, M., Chen, Y.F., How, J.P.: Collision avoidance in pedestrian-rich environments with deep reinforcement learning. arXiv preprint arXiv:1910.11689 (2019)

[14] Feddema, J.T., Lewis, C., , Schoenwald, D.A.: Decentralized control of cooperative robotic vehicles: Theory and application. IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION **18**(5), 852 – 864 (2002). https://doi.org/10.1109/TRA.2002.803466

[15] Foerster, J.N., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients. arXiv preprint arXiv:1705.08926 (2017)

[16] Fujimoto, S., van Hoof, H., Meger, D.: Addressing function approximation error in actor-critic methods. arXiv preprint arXiv:1802.09477 (2018)

[17] Godsil, C., Royle, G.: Algebraic Graph Theory. Springer-Verlag, New York, NY, USA (2000)

[18] Guo, Y., Hill, D.J., Wang, Y.: Nonlinear decentralized control of large-scale power systems. Automatica **36**(9), 1275–1289 (Sep 2000). https://doi.org/10.1016/S0005-1098(00)00038-8

[19] Haarnoja, T., Aurick Zhou, K.H., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., Levine, S.: Soft actor-critic algorithms and applications. arXiv preprint arXiv:1812.05905 (2018)

[20] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290 (2018)

[21] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv preprint arXiv:1801.01290 (2018)

[22] Han, M., Tian, Y., Zhang, L., Wang, J., Pan, W.: h_∞ model-free reinforcement learning with robust stability guarantee. arXiv preprint arXiv:1911.02875 (2019)

[23] van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. arXiv preprint arXiv:1509.06461 (2015)

[24] Iqbal, S., Sha, F.: Actor-attention-critic for multi-agent reinforcement learning. arXiv preprint arXiv:1810.0291 (2019)

[25] Keviczky, T., Borrelli, F., Balas, G.J.: Decentralized receding horizon control for large scale dynamically decoupled systems. Automatica **42**, 2105–2115 (2006). https://doi.org/10.1016/j.automatica.2006.07.008

[26] Khalil, H.K.: Nonlinear Systems. Prentice Hall, 3rd edition edn. (2001)

[27] Kiran, B.R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A.A.A., Yogamani, S., Pérez, P.: Deep reinforcement learning for autonomous driving: A survey. arXiv preprint arXiv:2002.00444 (2020)

[28] Levine, S.: Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint arXiv:1805.00909 (2018)

[29] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015)

[30] Lin, J., Morse, A., Anderson, B.: Lenient learners in cooperative multiagent systems. In: Proceedings of 42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475). Maui, HI, USA (Dec 2003). https://doi.org/10.1109/CDC.2003.1272825

[31] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. arXiv preprint arXiv:1706.02275 (2018)

[32] Opt: Scalable deep reinforcement learning for vision-based robotic manipulation, Q.: Dmitry kalashnikov and alex irpan and peter pastor and julian ibarz and alexander herzog and eric jang and deirdre quillen and ethan holly and mrinal kalakrishnan and vincent vanhoucke and sergey levine. arXiv preprint arXiv:1806.10293 (2018)

[33] Mayne, D., Rawlings, J., Rao, C., P.O.M.Scokaert: Constrained model predictive control: Stability and optimality. Automatica **36**(6), 789 – 814 (Jun 2000). https://doi.org/10.1016/S0005-1098(99)00214-9

[34] Olfati-Saber, R., Shamma, J.: Consensus filters for sensor networks and distributed sensor fusion. In: Proceedings of 44-th IEEE International Conference on Decision and Control. Seville, Spain, Spain (Dec 2005). https://doi.org/10.1109/CDC.2005.1583238

[35] Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE **95**(1) (Mar 2007). https://doi.org/10.1109/JPROC.2006.887293

[36] Ren, W.: Distributed leaderless consensus algorithms for networked Euler-Lagrange systems. International Journal of Control **82**(11), 2137–2149 (Nov 2009). https://doi.org/10.1080/00207170902948027

[37] Ren, W., Beard, R.W.: Decentralized scheme for spacecraft formation flying via the virtual structure approach. Journal of Guidance, Control, and Dynamics **27**(1), 706–716 (Jan/Feb 2004). https://doi.org/10.2514/1.9287

[38] Rezaee, H., Abdollahi, F.: A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots. IEEE Transactions on Industrial Electronics **61**(1), 347–354 (Jan 2014). https://doi.org/10.1109/TIE.2013.2245612

[39] Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P.: Trust region policy optimization. In: Proceedings of the 31 st International Conference on Machine Learning. pp. 1889–1897. Lille, France (Jun 2015)

[40] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)

[41] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introductions. The MIT Press, 2nd edn. (2018)

[42] Vásárhelyi, G., Virágh, C., Somorjai, G., Nepusz, T., Eiben, A.E., Vicsek, T.: Optimized flocking of autonomous drones in confined environments. Science Robotics **3**(20) (2018). https://doi.org/10.1126/scirobotics.aat3536

[43] Wang, J., Xin, M.: Integrated optimal formation control of multiple unmanned aerial vehicles. IEEE Transactions on Control Systems Technology **21**(5), 1731–1744 (Sep 2013). https://doi.org/10.1109/TCST.2012.2218815

[44] Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., Wang, J.: Mean field multi-agent reinforcement learning. arXiv preprint arXiv:1802.05438 (2018)
[45] Zhang, Q., H.T.Liu, H.: Aerodynamic model-based robust adaptive control for close formation flight. Aerospace Science and Technology **79**, 5–16 (Aug 2018). https://doi.org/10.1016/j.ast.2018.05.029
[46] Zhang, Q., Liu, H.H.T.: UDE-based robust command filtered backstepping control for close formation flight. IEEE Transactions on Industrial Electronics **65**(11), 8818–8827 (Nov 2018). https://doi.org/10.1109/TIE.2018.2811367, early access online, March 12, 2018
[47] Zhang, Q., Pan, W., Reppa, V.: Model-reference reinforcement learning control of autonomous surface vehicles with uncertainties. arXiv preprint arXiv:2003.13839 (2020)
[48] Zhang, Q., Pan, W., Reppa, V.: Model-reference reinforcement learning for collision-free tracking control of autonomous surface vehicles. arXiv preprint arXiv:2008.07240 (2020)
[49] Ziebart, B.D.: Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy (12 2010). https://doi.org/10.1184/R1/6720692.v1, https://kilthub.cmu.edu/articles/Modeling_Purposeful_Adaptive_Behavior_with_the_Principle_of_Maximum_Causal_Entropy/6720692