Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-2 (2020) 8150–8156

Multi-agent Formation Control with Obstacles Avoidance under Restricted Communication through Graph Reinforcement Learning

Huimu Wang*, **, Tenghai Qiu**, Zhen Liu**,
Zhiqiang Pu*, **, Jianqiang Yi*, **

** Scholl of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, 100049, China (e-mail: wanghuimu2018@ia.ac.cn, tenghai.qiu@ia.ac.cn, liuzhen@ia.ac.cn, zhiqiang.pu@ia.ac.cn, jianqiang.yi@ia.ac.cn).*
*** Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China*

Abstract: Multi-agent formation control with obstacles avoidance (MAFC-OA) is one of the attractive tasks of multi-agent cooperation. Although a number of algorithms can achieve formation control effectively, they ignore the nature structure feature of the graph formed by agents. Given this problem, a model, MAFC-OA, which is composed of observation attention network, action attention network and Multi-long short-term memory (Multi-LSTM) is proposed. With MAFC-OA, the agents can be trained to form the desired formation and avoid dynamic obstacles in the environments with restricted communication. Specifically, the above two attention networks not only incorporate the influence of the nearby agents' observation and actions, but also enlarge the agents' receptive field (communication range) through the chain propagation characteristics to promote cooperation among agents. Moreover, the Multi-LSTM allows the agents to take obstacles into consideration in the order of distance and to avoid the obstacles effectively. Simulations demonstrate that the agents can form the desired formation and avoid dynamic obstacles effectively.

Keywords: Reinforcement learning control, Multi-agent system, Deep reinforcement learning, Graph attention network, Formation control, Obstacles avoidance

1. INTRODUCTION

Cooperative multi-agent systems have been widely applied in fields as varied as smart grid control (Radhakrishnan et al. (2016)), autonomous driving (Shalev-Shwartz et al. (2016)), resource management (Zhang et al. (2019)), and robotics (Hang et al. (2016)). During these fields above, multi-agent formation control is one of the most attractive tasks that can be considered as a multi-agent cooperation problem.

For the formation control, there are two crucial issues including formation maintenance and collision avoidance (Sui et al. (2019)). For the former, the difficulty lies in how agents can maintain the expected formation in different situations. For the latter, the difficulty lies in how agents obtain all the information of environments or escape from the local minimum of force field ((Khatib et al. (1986))).

There are many attempts which focus on the multi-agent formation control problem where deep reinforcement learning (DRL) is one of the most popular method. It's worth noting that DRL has recently made exciting progress ((Mnih et al. (2015), Silver et al. (2016), Lillicrap et al. (2016)). Owing to the huge potential indicated by these deep learning based approaches, DRL has naturally been applied to the multi-agent reinforcement learning (MARL) to solve the multi-agent cooperation problems. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) is proposed in mixed multi-agent cooperative-competitive environments (Lowe et al. (2017)). It follows a common paradigm centralized learning with decentralized execution. Counterfactual Multi-agent (COMA) also utilzies a centralized critic and computes a counterfactual advantage function which handle the problem of multi-agent credit assignment by marginalizing the effect of each agent's action (Foerster et al. (2018)). However, the centralized critic used in MADDPG and COMA has to use the global state, while only partial observability (due to limited range or noisy sensors) and limited communication are available in most real-world environments, which means that each agent has to learn to behave cooperatively only on local observations and limited communication. To deal with this problem, the method of Mean-Field is proposed (Yang et al. (2018)) to capture the interaction of agents by mean action, which means that it can be applied to the environment with a large number of agents. Nevertheless, Mean-Field treats all agents' observation equally and ignores the fact that different agent's observation has different influences on the centralized critic. To address these problems, the attention mechanism (Oh et al. (2016), Vaswani et al. (2017)) has been employed to effectively represent the influence of agents on centralized critics. Multiple-Actor-Attention-Critic (MAAC) employs the attention mechanism to model a centralized critic, from which decentralized actors are derived using soft actor-critic (Iqbal et al. (2019)). But it still needs to take all agents' states into consideration, which is unaviable in reality. Besides, the requirement of all agents' states makes the MAAC not generalizable. However, graph networks have an inherent advantage to describe the communication topology among the agents. The characteristic of information exchange which is expressed in the graph could potentially help aggerate all agent's observation via long chain and promote the cooperation behaviour (Wu et al. (2019)). Nonetheless, none of these approaches above studies multi-agent cooperation from the graph perspective and ignores the structure of multi-agent system.

Motivated by these issues, we propose a model named MAFC-OA building on the framework of Graph Attention Networks (GAT) (Petar et al. (2018)). The model can be divided into two parts including formation control and obstacles avoidance. The first part based on the GAT focuses on the cooperation behaviour among the agents. A double graph attention network including agent observation attention network and agent action attention network is designed for training the cooperation behaviour. It not only incorporates the influence of nearby agents' observation and actions, but also enlarges the agents' receptive field or communication field through the chain propagation characteristics of graph neural networks to promote cooperation among the agents. The second part focuses on the obstacle avoidance. Instead of using the attention network, we extend long short-term memory (LSTM) method (Everett et al. (2018)) as Multi-LSTM in multi-agent environments. The Multi-LSTM allows the agents to take obstacles into consideration in the order of distance and avoid arbitrary number of obstacles. Moreover, in order to scale up to more agents, we use the parameters sharing method to train all the agents in a decentralized framework. We evaluate our architecture in environments with different number of agents, and the simulation results demonstrate that the agents can form the expected formation and avoid obstacles under the proposed model.

2. PRELIMINARIES

2.1 Problem Statement

The environment of the multi-agent formation control with obstacles avoidance based on Multi-Agent Particle Environment (MAPE) (Lowe et al. (2017)) is presented in Fig. 1. As shown in Fig. 1(a), six agents represented by blue circles are generated randomly in a 2×2 square map of MAPE. The formation centre represented by a red star follows a random trajectory in each episode. Besides, the target centre represented by a red triangle is generated randomly in the map. Moreover, there are several obstacles represented by dark circles. They are generated randomly near the map edge and can move in random velocities. When the agents pass through the environment, each agent can only observe its own state and the obstacles in a fixed range. Moreover, considering the limitation of communication bandwidth in reality, each agent can only communicate with up to two nearby agents in this paper. In this paper, it is expected that the agents can avoid the dynamic obstacles, be evenly distributed around the moving formation centre and arrive at the target point .

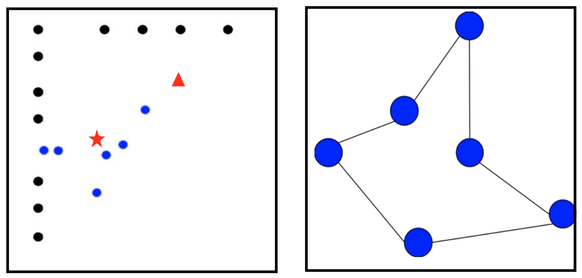
(a) Agents and obstacles (b) Graph communication

Fig. 1. Simulation Environment

Let S_t denote the environment state at time t and o_i^t denote the agent i 's local observation including its position, velocity, the formation center position and the obstacles' positions. Specifically, we assume that at time t , the agent i 's position $p_i^t=[p_i^{t^x},p_i^{t^y}]$, the agent i 's velocity $v_i^t=[v_i^{t^x},v_i^{t^y}]$,the formation center position $p_c^t=[p_c^{t^x},p_c^{t^y}]$ and the obstacle j 's position $p_{oj}^t=[p_{oj}^{t^x},p_{oj}^{t^y}]$. Besides, the action space for each agent is discretized. The agent can move one step (0.1units) in both X and Y directions.

Moreover, as shown in Fig. 1(b), the information communication among the agents can be represented in undirected graph $G=(V,E)$. Specifically, $V=\{1,\ldots,N\}$ denotes the nodes consisting of the agents. $E\subseteq V\times V$ denotes the edge set consisting of communication status among the agents where an edge from node i to j is denoted as $(i,j)\in E$ .Besides, h is a set of node features, $h=\{\vec{h}_1,\vec{h}_2,\ldots,\vec{h}_N\}$, $\vec{h}_i\in\mathbb{R}^F$,where F is the number of features in each node. Moreover, N_i is a set of neighbours communicating with node i in the graph. There is a weighted adjacency matrix A where $a_{ij}=1$ if $j\in N_i$ otherwise $a_{ij}=0$.

2.2 Reinforcement Learning

The environment in this paper is regarded as Partially Observable Markov Games which is an extension of the framework of Markov Games (Littman et al. (1994)). It is defined by a global state S, a set of actions $A_1,\ldots,A_N$, and a set of local observations $O_1,\ldots,O_N$. The local observation is composed of the agent's position p_i^t and velocity v_i^t , the formation center position p_c^t, and the obstacles' positions p_{oj}^t. To choose actions, each agent uses a learnable policy $\pi_i: O_i\rightarrow P_a(A_i)$,which produces the next state according to the state transition function $T: S\times A_1\times\ldots\times A_N\rightarrow P_t(S')$ that defines the probability distribution over possible next states, given current states and actions for each agent. Each agent obtains rewards from the environment after all agents take

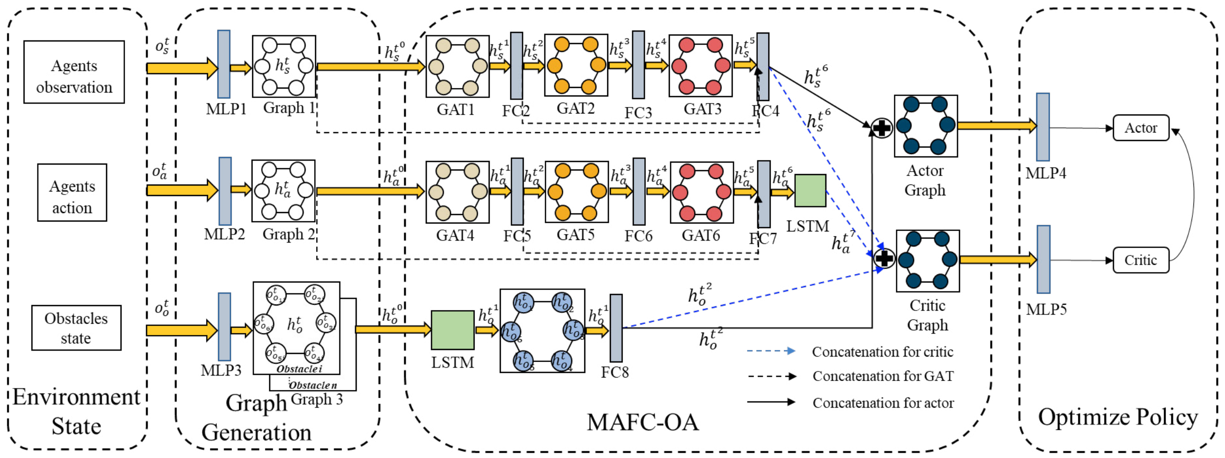

Fig. 2. Structure of MAFC-OA

actions: $R_i = S \times A_1 \times \ldots \times A_N \rightarrow \mathbb{R}$. The agents aim to learn a policy that maximizes their expected discounted returns,

$$J_i(\pi_i) = E_{a_1 \sim \pi_1, \ldots, a_N \sim \pi_N, s \sim T}\left[\sum_{t=0}^{\infty} \gamma^t r_{it}\left(s_t, a_{1t}, \ldots, a_{Nt}\right)\right] \quad (1)$$

where r_{it} is the reward that agent i obtain at time t, s_t represents the global state S at time t. $\gamma \in [0,1]$ is the discount factor that determines how much the policy favors immediate reward over long-term gain.

2.3 Policy Gradient(PG)

Policy gradient (Sutton et al.(2000)) aims to estimate the gradient of a single agent's expected returns $J_i(\pi_i)$with respect to the parameters (θ) of its policy(π_θ). This gradient estimate takes the following form:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \log\left(\pi_\theta\left(a_t \middle| s_t\right)\right) \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}\left(s_{t'}, a_{t'}\right) \quad (2)$$

2.4 Proximal Policy Optimization (PPO)

Most PG methods perform one gradient update per sampled trajectory, which results in high sample complexity. The PPO algorithm is proposed by Schulman et al. (2017) to address the problem. Let

$$l_t(\theta) = \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta^k}(a_t | s_t)}$$

denote the likelihood ratio. $\pi_{\theta k}$ represents the agent policy before K steps. Then PPO optimizes the objective

$$\begin{aligned} L(\theta) = E[\min(l_t(\theta)\hat{A}_t^{\theta^k}(s_t, a_t), \\ clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(s_t, a_t)] \end{aligned} \quad (3)$$

where $\hat{A}_t^{\theta^k}(s_t, a_t)$ is the generalized advantage estimate and $clip\left(l_t(\theta), 1-\varepsilon, 1+\varepsilon\right)$ clips $l_t(\theta)$ in the interval $[1-\varepsilon, 1+\varepsilon]$.

3. APPROACH

In order to train these agents to maintain a formation and avoid dynamic obstacles, a graph reinforcement learning model MAFC-OA is proposed as shown in Fig. 2. The model can be divided into two parts, formation control and obstacle avoidance.

The first part focuses on the cooperation behaviour among the agents. A double graph attention network including agent observation attention network and agent action attention network is designed for training the cooperation behaviour. The observation attention network has three layers to conduct information exchange among the agents. Furthermore, the action attention network has the similary structure as the observation attention network to solve the non-stationary problem of the multi-agent system. The details of the two graph attention networks are presented in section 3.1 and 3.2.

The second part focuses on the problem of obstacle avoidance. The LSTM is extended (Everett et al. (2018)) as a Multi-LSTM in multi-agent environments. The Multi-LSTM is constructed for two reasons as follows. On one hand, the variant number of obstacles results in the change of the adjacency matrix in its dimension and value, which means that it is difficult to converge in the process of training due to the dynamic graph. On the other hand, the direct use of attention network could not improve the model's ability, and the network may also be difficult to converge due to a mass of hyperparameters resulting by complex network components. Therefore, the Multi-LSTM is designed to avoid obstacles. The details of the Multi-LSTM are presented in section 3.3.

3.1 Agent Observation Attention Network

Intuitively, each agent requires all the other agents' information about their observations and actions to behave cooperatively. However, it is impractical for each agent to get information of all the other agents due to the limitation of communication. Therefore, to obtain the information of the other agents which are not communicate with the agent directly, the attention network based on graph convolution is designed due to the chain propagation characteristics of the graph convolution. By stacking graph convolutional layers, the receptive field of an agent gradually grows. Then the agent can collect more information, which means that the scope of cooperation can be expanded. Therefore, three graph convolution layers are utilized to enlarge each agent's reception field. Under this setting, the number of the agents that each agent can communicate with can be increased from two to six.

3.1.1 Graph Generation

Given the analysis above, a graph $G=(V,E)$ is generated before the graph attention network is designed to aggregate the agents' information. As shown in Fig. 2, the agents' observations o_s^t is embedded as node feature h of the graph $G=(V,E)$ via one-layer MLP1 represented by g_i:

$$h_i^t = g_i(o_i^t) = \sigma(o_i^t W_M^1 + b_M^1) \tag{4}$$

where h_i^t is one of the node feature h of the graph $G=(V,E)$, $o_i^t\left(o_i^t \in o_s^t\right)$ represents the local observation of agent i at time t, W_M^1 and b_M^1 are weight matrix and bias of MLP1 to learn, σ represents ReLu function.

Then we define a symmetric adjacency matrix $A \in \mathbb{R}^{N\times N}$ according to the communication status among the agents. It is noted that the value of a_{ij} ($a_{ij} \in A$) is varied with the change of distance between the agents. Besides, there is a self-loop for each agent because the cooperation behaviour is decided not only by its neighbourhoods' observation but also by its own observation.

3.1.2 Attention Distribution

After the graph is generated, the nearby agent's observation can be taken into consideration to form a designated formation by stacking the graph convoluaiton layers. In addtion to enlarging the receptive fields of agents, the neighbour agents need to be treated differently by agent *i* for promoting cooperation. One of the rest agents may be farther away from agent *i* than the other agents, which means agent *i* will be influenced differently due to the different distance among agents.

Therefore, the GAT layers are leveraged to allow each agent to treat the other agents' states differently and enlarge agents' reception field. To aggregate the state from agent *j* in determining the policy for agent i, we embed the hidden state of the two agents from the MLP1 and calculate the attention coefficients e_{ij} from agent *j* to agent i and its normalized form:

$$e_{ij} = a_G^k\left(W_G^k h_i, W_G^k h_j\right) \tag{5}$$

$$\alpha_{ij} = \mathrm{softmax}(e_{ij}) == \frac{\exp\left(\mathrm{LeakyReLU}\left(e_{ij}\right)\right)}{\sum_{k\in N_i}\exp\left(\mathrm{LeakyReLU}\left(e_{ij}\right)\right)} \tag{6}$$

where a_G^k is a single-layer feedforward neural network, parametrized by a weight vector and W_G^k is a learnable weight matrix, LeakyReLU is a nonlinear activation function. Once we obtain the attention coefficient α_{ij}, the latent node features $h_s^{t'}$ can be computed by a liner combination of neighbourhood hidden states as shown in (7):

$$h_s^{t'} = \sigma\left(\sum\nolimits_{j\in N_i} \alpha_{ij} W h_s^t\right) \tag{7}$$

$$h_s^{t'} = \Big\|_{m=1}^{K} \sigma\left(\sum\nolimits_{j\in N_i} \alpha_{ij}^m W^m h_s^t\right) \tag{8}$$

where $\|$ represents the concatenation , K represents the number of heads, α_{ij}^m represents the normalized attention coefficient of the m*th* attention mechanism and W^m represents the weight matrix of the m*th* linear transformation.

Vaswani et al. (2017) and GAT (2018) indicate that the import of multi-head attention is beneficial to stabilize the learning process of the attention. Besides, the agent can different state representation of the nearby agents from different representation subspace with multi-head setting. Thus, Equation (8) is utilized to obtain the hidden state $h_s^{t'}$ under the setting of multi-head attention.

Besides, after the final GAT layer, the hidden state $h_s^{t^0}$ and $h_s^{t^2}$ are concatenated with $h_s^{t^5}$ and they are fed into FC4 as shown in (9). Since the hidden state could disappear during the process of graph convolution, these hidden states are concatenated in the final layer to stabilize the training process

$$h_s^{t^5} = \sigma\left(\left[h_s^{t^5} \middle\| h_s^{t^2} \middle\| h_s^{t^0}\right] W_F^4 + b_F^4\right) \tag{9}$$

3.2 Agent Action Attention Network

The multi-agent cooperation behaviour is determined not only by the nearby agents' observation, but also the nearby agents' actions. As mentioned in MADDPG, the multi-agent environment can be treated as stationary if we know the actions taken by all the agents, since

$$\begin{aligned} P(s'|s,a_1,\dots,a_N,\pi_1,\dots,\pi_N) &= P(s'|s,a_1,\dots,a_N) \\ &= P(s'|s,a_1,\dots,a_N,\pi_1',\dots,\pi_N') \end{aligned} \tag{10}$$

Hence, the actions of as many agents as possible shoule be taken into account. In our environments, the access to other agents' action is also limited by the communication distance or the communication bandwidth. Therefore, a network structure is designed to enlarge the respective field and to treat the other agents' states for each agent. As shown in Fig. 2, after passing through 3-layer GAT, the hidden state obtained is as follows:

$$h_a^{t^5} = \sigma\left(\left[h_a^{t^5} \middle\| h_a^{t^2} \middle\| h_a^{t^0}\right] W_F^7 + b_F^7\right) \tag{11}$$

Besides, in order to better consider the actions of other agents in the process of learning, a LSTM layer is adopted to process current and previous actions.

3.3 Obstacles Avoidance

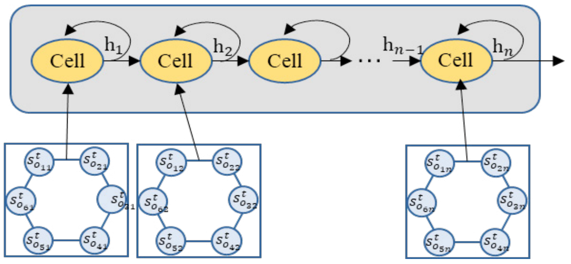

Fig. 3. Multi-LSTM

During the process of obstacles avoidance, there are two crucial facts need to be considered—the variant number of obstacles in the detective range and different influence of different obstacles. To be specific, the number of obstacles in the agent's detective range is variant due to the moving agents and obstacles. The varying obstacles lead to an un-fixed input for feedforward neural networks. In the graph network, it means that the topology is dynamic. If we utilize graph convolution layer, the adjacency matrix will change in its dimension and value. The change means the graph of the obstacles and agents is dynamic, which will result in difficulty of convergence. To address these problems, a LSTM is adopted to process the obstacles' state. In general, the LSTM is a recurrent architecture that the output is produced from a combination of a stored cell state and an input. It can accept an arbitrary-length sequence to produce a fixed-size output. In other words,this advantageous property solves the problem of unfixed size of input. Based on the characteristics, Everett et al. (2018) utilizes the LSTM to encode a sequence of information that is not time-dependent.

In the case of a large number of agents, the early states have less influence on the agent, so the states are fed in reverse order of distance to the agent, which means that the closest agents (fed last) should have the biggest effect on the final hidden state. In other words, this is a similar way to the attention mechanism but it's simpler. The importance of obstacles to the agents only depends on the distance between them instead of the attention coefficients. In this way, the problem of variant number of obstacles and different influence of different obstacles can be solved. Therefore, the existing LSTM (Everett et al. (2018)) is extended to multi-agent systems instead of using the GAT. As shown in Fig. 3, all the agents share the LSTM layer, $s^t_{o_{ij}}$ represents the relative position of obstacle j to agent i at time t. For instance, the Multi-LSTM receives $s^t_{o_{i1}}$ then generates h_1, then feeds h_1 and relative distances between obstacle 2 and agents i to produce h_2, and so on. Finally, LSTM outputs the last hidden state h_n after all the obstacles and all the agents have been processed. It is intuitive that h_n contains encoded information about all the obstacles.

3.4 Components Integration

After the states are extracted by the MAFC-OA model and the Multi-LSTM, they are utilized to optimize the agents' policy. For the agents' learning, the centralised training and decentralised execution is a common paradigm. However, there are two disadvantages for the paradigm. On one hand, a critic is learned with information from all agents, which is not available in the reality. On the other hand, the actors receive information only from their nearby agents, which influences the performance of cooperation. As mentioned above, these disadvantages can be addressed with the MAFC-OA model. In particular, although a critic is learned with information from nearby agents in this paper, it can obtain all the other agents' information by the MAFC-OA model. Besides, the actor can receive all the other agents' information to promote the cooperation behaviour. Thus, the centralised training and decentralised execution can be adopted with the MAFC-OA model to circumvents the challenge of non-Markovian and non-stationary environments during learning. Through the above analysis, we design a complete architecture including graph generation, the MAFC-OA model and policy optimization in Fig. 2.

As shown in Fig. 2, all the agents' information including observations, actions and obstacle states are extracted with the MAFC-OA model as $h_s^{t^6}$, $h_a^{t^7}$ and $h_o^{t^2}$. Then $h_s^{t^6}$ and $h_o^{t^2}$ are concatenated as the inputs to actor network.

$$h^{actor}_{total} = \sigma\left(\left[h_s^{t^6} \middle\| h_0^{t^2}\right] W_M^4 + b_M^4\right)$$

$h_s^{t^6}$, $h_o^{t^2}$ and $h_a^{t^7}$ are concatenated as the inputs to critic network.

$$h^{critic}_{total} = \sigma\left(\left[h_a^{t^7} \middle\| h_s^{t^6} \middle\| h_0^{t^2}\right] W_M^5 + b_M^5\right)$$

After these inputs are obtained, the PPO is implemented in an actor-critic framework. According to the objective function of PPO as shown in (3), it is changed as (12) after the concatenation of all the states:

$$l_t(\theta) = \frac{\pi_\theta(a_t | h^{actor}_{total})}{\pi_{\theta^k}(a_t | h^{actor}_{total})}$$

$$\begin{aligned} L(\theta) = E[&\min(l_t(\theta)\hat{A}_t^{\theta^k}(h^{critic}_{total}),\\ &clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(h^{critic}_{total})] \end{aligned} \tag{12}$$

Moreover, to scale up to more agents, the parameters sharing method is applied to train all the agents in a decentralized framework.

4. SIMULATION AND RESULTS

4.1 Environments settings

In order to demonstrate and evaluate our proposed architecture, 4 environments are designed in this paper. As Fig. 4 shows, scenario (a) and (b) are designed to evaluate the effectiveness of our architecture. The only difference between them is the number of agents. The red star represents the target point that the agents need to keep the formation to arrive. Scenario (c) and (d) are designed to evaluate the effectiveness of obstacle avoidance. The agents need to form the expected formation without collision. After forming the formation, the agents need to keep the formation, pass through the group of obstacles

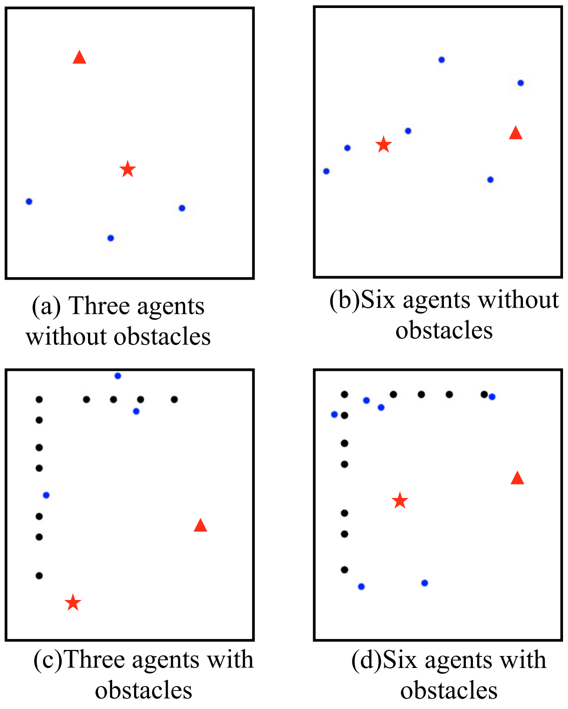
(a) Three agents without obstacles
(b)Six agents without obstacles
(c)Three agents with obstacles
(d)Six agents with obstacles

Fig. 4. Environment scenarios

without any collision and arrive at the target point.

4.2 Simulation results

The most recent work which is similar to us is (Akshat et al. (2019)). Although it focuses on multi-agent formation control but does not consider the behaviour of obstacle avoidance. We compare our architecture with it in the 4 environments.

First, the simulation results conducted in the environments with 3 agents are presented in Fig. 5, curves of min distances between the agents and the expected positions, steps, distance entropy, loss of value and success rate are given as indicators of our training in the environments with 3 agents. Specifically,

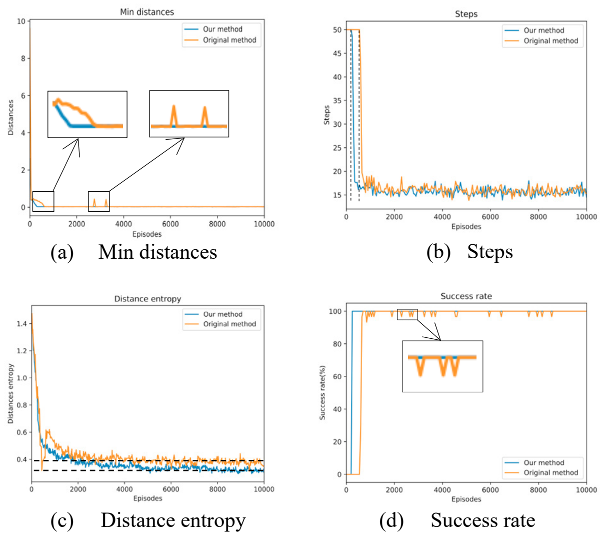

Fig. 5. Training results of 3 agents

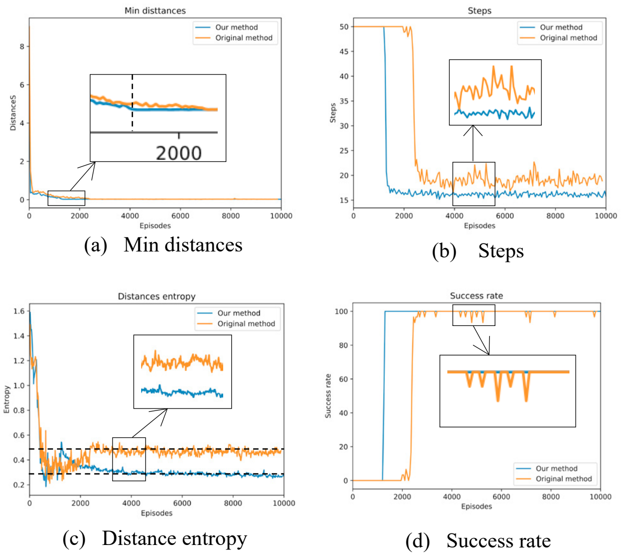

Fig. 6. Training results of 6 agents

the curve of min distance reflects whether the agents can form the designated formation correctly. Moreover, the curves of distance entropy and steps reflect the time the agents need to take to form the designated formation. Besides, the curve of success rate reflects obviously the performance of the training methods. It can be observed from Fig. 5 that all the curves converge to a stable status, which demonstrate the effectiveness of the proposed model. Besides, all the curves reflect that the proposed model converges faster than the original method. The fluctuation of the curves of steps and distance entropy is because that the agents generated randomly in the map have different positions in each episode. As indicated by the curves of success rate, there are some fluctuation with the original method. On the contrary, there are no fluctuation with the proposed method, which indicates that the proposed method is more stable than the original method.

Next, the curves of training in the environments with 6 agents are presented in Fig. 6. There is no obvious difference in the curve of min distance because the min distance is depended on the formation demand. The curves of steps and success rate demonstrate that the proposed method performs better when the number of agents increases. Besides, the curve of distance entropy indicates that the proposed method has more robust performance.

Table 1. Success rate with obstacles avoidance

Number of obstacles	1	3	6	12
The original method (3)	100/100	95/100	88/100	82/100
The original method (6)	97/100	90/100	87/100	80/100
The proposed method (3)	100/100	98/100	96/100	95/100
The proposed method (6)	99/100	97/100	95/100	94/100

Moreover, our proposed method is compared with the original method in the environments with different number of obstacles. The results can be observed in TABLE 1. It is shown that the improvement of our method over the original method grows with respect to the number of obstacles. The performance of obstacles avoidance of the original method gradually declining when the number of agents grows. In contrast, our method can keep the satisfying performance of obstacles avoidance regardless of the rise in the number of obstacles and agents.

5. CONCLUSIONS

In this paper, we present a MAFC-OA model for multi-agent formation control with obstacle avoidance under restricted communication. The key idea is to utilize the characteristics of graph, attention and LSTM to promote the cooperation behavior. The model is shown to perform a satisfying strategy under with dynamic obstacles. Future work will expand the fixed number of agents to variant number of agents and consider the spatial temporal characterises of multi-agent systems.

ACKNOWLEDGEMENT

This work is supported by the grants of National Key Research and Development Program of China (No. 2018AAA0102404), and Innovation Academy for Light-duty Gas Turbine, Chinese Academy of Sciences (No. CXYJJ19-ZD-02).

REFERENCES

Agarwal, A., Kumar, S., and Sycara, K. (2019). Learning transferable cooperative behavior in multi-agent teams. arXiv preprint arXiv:1906.01202.

Everett, M., Chen, Y.F., and How, J.P. (2018). Motion planning among dynamic, decision-making agents with deep reinforcement learning. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3052–3059. IEEE.

Foerster, J.N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018). Counterfactual multi-agent policy gradients. In Thirty-Second AAAI Conference on Artificial Intelligence.

Iqbal, S. and Sha, F. (2019). Actor-attention-critic for multi-agent reinforcement learning. In International Conference on Machine Learning, 2961–2970.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. In Autonomous robot vehicles, 396–404. Springer.

Li, X., Zhang, J., Bian, J., Tong, Y., and Liu, T.Y. (2019). A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network. In Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, 980–988. International Foundation for Autonomous Agents and Multiagent Systems.

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

Littman, M.L. (1994). Markov games as a framework for multi-agent reinforcement learning. In Machine learning proceedings 1994, 157–163. Elsevier.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O.P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. In Advances in Neural Information Processing Systems, 6379–6390.

Ma, H., Tovey, C., Sharon, G., Kumar, T.S., and Koenig, S. (2016). Multi-agent path finding with payload transfers and the package-exchange robot-routing problem. InThirtieth AAAI Conference on Artificial Intelligence

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529.

Oh, J., Chockalingam, V., Singh, S., and Lee, H. (2016). Control of memory, active perception, and action in minecraft. In Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48, 2790–2799. JMLR. org.

Radhakrishnan, B.M. and Srinivasan, D. (2016). A multiagent based distributed energy management scheme for smart grid applications. Energy, 103, 192–204.

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

Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving. arXiv preprint arXiv:1610.03295.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. nature, 529(7587), 484.

Sui, Z., Pu, Z., Yi, J., and Xiong, T. (2019). Formation control with collision avoidance through deep reinforcement learning. In 2019 International Joint Conference on Neural Networks (IJCNN), 1–8. doi: 10.1109/IJCNN.2019.8851906.

Sutton, R.S., McAllester, D.A., Singh, S.P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In Advances in neural information processing systems, 1057–1063.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems, 5998–6008.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In International Conference on Learning Representations. URL https://openreview.net/forum?id=rJXMpikCZ.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P.S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Yang,Y.,Luo,R.,Li,M.,Zhou,M.,Zhang,W.,andWang, J.(2018). Mean field multi-agent reinforcement learning. In International Conference on Machine Learning, 5567–5576.