An Enhanced Model-Free Reinforcement Learning Algorithm to Solve Nash Equilibrium for Multi-Agent Cooperative Game Systems

YUANNAN JIANG AND FUXIAO TAN
College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China
Corresponding author: Fuxiao Tan (fuxiaotan@gmail.com)

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 61673117 and Grant 52071200.

ABSTRACT Solving the Nash equilibrium is important for multi-agent game systems, and the speed of reaching Nash equilibrium is critical for the agent to quickly make real-time decisions. A typical scheme is the model-free reinforcement learning algorithm based on policy iteration, which is slow because each iteration will be calculated from the start state to the end state. In this paper, we propose a faster scheme based on value iteration, using Q-function in an online manner to solve the Nash equilibrium of the system. Since the calculation is based on the value from the last iteration, the convergence speed of the proposed scheme is much faster than the policy iteration. The rationality and convergence of this scheme are analyzed and proved theoretically. An actor-critic network structure is used to implement this scheme through simulation. The simulation results show that the convergence speed of our proposed scheme is about 10 times faster than that of the policy iteration algorithm.

INDEX TERMS Nash equilibrium, multi-agent game systems, model-free reinforcement learning.

I. INTRODUCTION

MULTI-AGENT consensus research involves the knowledge, goals, skills, and planning of how to enable the agents to take coordinated actions to solve problems. Multi-agent consensus mainly studies how to design algorithms to make all agents in a specified state reach an optimal control policy by independently optimizing each agent’s performance index based on the Hamilton-Jacobi-Bellman equation, whose result is the Nash equilibrium solution [1]–[3]. So that game theory can provide a framework for multi-agent consensus research [4], [5]. References [6], [7] proposed the concept of cooperative Nash equilibrium, with which, the dynamics and value function of each agent depend only on the actions of the agent and its neighbors. The graphical game can provide Nash equilibrium solutions among neighbors.

The research on the consensus problem began in the 80’s. Traditional methods such as natural animal group models, Boid model, and Vicsek model are inspired by the movement rules of nature [8]–[10]. In recent years, reinforcement learning (RL) is one of the areas that have attracted the most research and development interest. RL maps the relationship between the learning state and behavior of agents, involving how for agents to choose their behavior in a dynamic environment to optimize the sum of cumulative rewards [11]–[13]. Many algorithmic ideas of RL can be applied to the consistency research of multi-agents [7], [14], [15]. Multi-agent RL algorithms can be model-based or model-free [16], [17], where models are used to represent environments. Normally, models can be very helpful for the agent to deal with various situations and choose the best action [18]. However, the model building process requires a lot of information and time [19]. If the environment is unknown, possible subsequent states of the current state cannot be known. In this case, it is impossible to determine the best action to be taken in the next state. Therefore, the model-free algorithm is an important research direction for multi-agent systems used in unknown environments [20], [21]. Such kind of algorithms include those proposed in references [16], [22], which solve the related Nash equilibrium based on policy iteration, working as follows. It starts with an initial policy, and then evaluates and improves the policy, then further evaluates & improves the improved policy. After continuous iteration and updating, the policy will be optimized. Since the evaluation of the policy in each iteration is calculated from the start state to the end state, it needs a lot of time to get the best policy and...
optimal value when the state space is large [11]. The policy iteration algorithm proposed in reference [23] has relatively strong learning ability by adding a sub-iteration for iterative performance index functions.

In this paper, we propose a value iteration algorithm to solve the Nash equilibrium for multi-agent game systems by designing a cooperative agent’s RL algorithm jointly using Q-function in an online manner. For value iteration, its working steps are basically the same as those of policy iteration. However, with the iteration of the state value, the policy is adjusted in time to greatly reduce the number of iteration steps and improve the convergence speed. Value iteration guarantees that the value rather than the policy is optimal. In addition, policy iteration needs to formulate an initial policy to ensure convergence but not for value iteration [11]. The rationality and convergence of this algorithm are analyzed and proved theoretically. An actor-critic network structure is used to implement this algorithm. An actor-critic network includes an actor network and a critic network. The actor network uses the policy function to generate actions and interact with the environment. The critic network uses the value function to evaluate the performance of the actor network, and then guides the actor’s action in the next stage. The simulation results with MATLAB show that the convergence speed of the proposed algorithm is about 10 times faster than that of the policy iteration algorithm.

In comparison with the model-based algorithms proposed in [1], [6], the complexity of our model-free algorithm is relatively low, with no requirement of the knowledge of system dynamics. The simulation results in Section VI show the convergence speed of these algorithms.

Section I is Introduction. The remainder of this paper is organized as follows. Section II describes the synchronization control problem of the multi-agent system on the graph. Section III introduces some results about optimal control used to solve the dynamic graphical games proposed in this paper. Section IV discusses the existing Nash solution and the best response for the graphical games. Section V proposes an enhanced RL algorithm, which is implemented by a simulation study in Section VI. The paper is concluded in Section VII.

II. MULTI-AGENT GRAPHS AND SYNCHRONIZATION

This section introduces multi-agent graphs and synchronization problem to be addressed in this paper. Some available results used here and henceforth are summarized in Section III. The definitions of the main notations used in the discussion are listed in table 1:

| Variables | Definitions |
|-----------|-------------|
| A         | System matrix |
| B_i       | Input matrix of agent i |
| u_i       | Control input of agent i |
| u_{i-1}   | Control inputs of the neighbors of agent i |
| e_i       | Local neighborhood tracking error of agent i |
| e_{i-1}   | Local neighborhood tracking error of the neighbors of agent i |
| e_i       | Vector of e_i and e_{i-1} |
| η          | Synchronization error vector |

as \( D = \text{diag} \{d_i\} \), where, \( d_i = \sum_{j \in N_i} e_{ij} \) is the input-degree of node i. The Laplace matrix \( L \) is defined as \( L = D - E \).

B. STATE SYNCHRONIZATION AND ERRORS

A communication diagram is composed of N agents. The local dynamic system of agent i is defined as follows:

\[
x_i(k + 1) = Ax_i(k) + Bu_i(k),
\]

Consider a leader node \( v_0 \) that has command generator dynamics \( x_0(k) \in \mathbb{R}^n \), which is given by

\[
x_0(k + 1) = Ax_0(k).
\]

The consensus of multi-agent systems is to synchronize the states of the follower and leader agents, i.e.,

\[
\lim_{k \to \infty} ||x_i(k) - x_0(k)|| = 0,
\]

by designing related inputs \( u_i(k) \) and combining with the information from neighbor agents.

To study the synchronization problem on graphs, we define the local neighborhood tracking error \( \varepsilon_i(k) \in \mathbb{R}^n \) for the follower agent i as follows:

\[
\varepsilon_i(k) = \sum_{j \in N_i} e_{ij}(x_i(k) - x_j(k)) + g_i(x_0(k) - x_i(k)),
\]

where, \( g_i \geq 0 \) is the gain between the leader agent and follower agent i. If node i has a connection to the leader node, then \( g_i > 0 \) [6].

The overall tracking error vector is written as

\[
\varepsilon(k) = ((L + G) \otimes I_n) (x_0(k) - x(k)),
\]

where, \( x = [x_1^T, x_2^T, x_3^T, \ldots, x_N^T]^T \) is the global state vector, and \( \varepsilon = [\varepsilon_1^T, \varepsilon_2^T, \varepsilon_3^T, \ldots, \varepsilon_N^T]^T \) is the global tracking error vector. So (4) can be written as \( \varepsilon(k) = -((L + G) \otimes I_n) \eta(k) \).

The synchronization error vector is written as

\[
\eta(k) = (x(k) - x_0(k)) \in \mathbb{R}^{Nn},
\]

where, \( x_0 = Ix_0, I = 1 \otimes I_n, \) and 1 is the N-vector of ones. \( G = \text{diag} \{g_i\} \in \mathbb{R}^{N \times N} \) is a diagonal matrix of pinning gains. If the graph contains a spanning tree and \( g_i \neq 0 \) for the root node, then \((L + G)\) is nonsingular, i.e.,

\[
\|\eta(k)\| \leq \frac{\|\varepsilon(k)\|}{\sigma(L + G)^T},
\]
For easy identification, when the index time \( k \) is clear, \( x_i(k) \) can be written as \( x_{ik} \).

The next result shows that the synchronize error vector can be made arbitrarily small by making local neighbor tracking errors, whose dynamics for node \( i \) are given by

\[
e_i(k+1) = f_i(e_{ik}, u_{ik}, u_{-ik}) = A e_{ik} - (d_i + g_i) B_i u_{ik} + \sum_{j \in N_i} e_j B_j u_{jk}.
\]

(7)

Therefore, the work of solving the Nash equilibrium of multi-agent systems is to minimize (7) [1], [6]. Game theory and graph theory are used to explain the responses of agent \( i \) to other agents in multi-agent systems. To define a dynamic graphical game, the control inputs of the neighbors of agent \( i \) is defined as

\[
u_{-i} = \{u_j | j \in N_i \}.
\]

(8)

Then local performance index for agent \( i \) can be written as

\[
J_i = \sum_{k=0}^{\infty} U_i(e_{ik}, u_{ik}, u_{-ik}),
\]

(9)

where, \( U_i(e_{ik}, u_{ik}, u_{-ik}) \) is the utility function for agent \( i \), and written as

\[
U_i(e_{ik}, u_{ik}, u_{-ik}) = \frac{1}{2} \left( e_{ik}^T Q_{ii} e_{ik} + u_{ik}^T R_i u_{ik} + \sum_{j \in N_i} u_{jk}^T R_j u_{jk} \right).
\]

(10)

Given the policies, (10) catches the local information given by (7) for agent \( i \). The solution will be given in terms of the local neighbor tracking error (7).

### III. THE RELATED WORKS USED IN THE PROPOSAL

This section introduces some results of optimal control used to solve the dynamic graphical game proposed in this paper [6].

#### A. BELLMAN FUNCTION

The Bellman function is defined as

\[
V_\pi^o(\bar{e}_{ik}) = \min_{u_{ik}} \left( V_\pi^o(\bar{e}_{ik+1}) + \sum_{j \in N_i} u_{jk}^T R_j u_{jk} \right)
\]

(11)

The difference of the value function \( \Delta V_\pi^o(\bar{e}_{ik}) \) is defined as follows:

\[
\Delta V_\pi^o(\bar{e}_{ik}) = V_\pi^o(\bar{e}_{ik+1}) - V_\pi^o(\bar{e}_{ik}),
\]

(12)

and its gradient is

\[
\nabla V_\pi^o(\bar{e}_{ik+1}) = \frac{\partial V_\pi^o(\bar{e}_{ik+1}, u_{ik+1})}{\partial \bar{e}_{ik+1}}.
\]

(13)

The goal for the multi-agent graphical games is to find the optimal value for agent \( i \) as the minimum of the value function \( V_\pi^o(\bar{e}_{ik}) \) such that

\[
V_\pi^o(\bar{e}_{ik}) = \min_{u_{ik}} \left( V_\pi^o(\bar{e}_{ik+1}) + \sum_{j \in N_i} U_i(e_{ij}, u_{ij}, u_{-ij}) \right).
\]

(14)

Given stationary admissible polices for the neighbors of agent \( i \), the Bellman optimality principle yields

\[
V_\pi^o(\bar{e}_{ik}) = \min_{u_{ik}} \left( U_i(e_{ik}, u_{ik}, u_{-ik}) + V_\pi^o(\bar{e}_{ik+1}) \right),
\]

(15)

where,

\[
u_{ik}^0 = \arg \min_{u_{ik}} \left( U_i(e_{ik}, u_{ik}, u_{-ik}) + V_\pi^o(\bar{e}_{ik+1}) \right),
\]

and

\[
\frac{\partial V_\pi^o(\bar{e}_{ik})}{\partial u_{ik}^o} = 0.
\]

Then

\[
\pi_{ik} = u_{ik}^o = M_i \nabla V_\pi^o(\bar{e}_{ik+1}),
\]

(16)

and set \( M_i = R_i^{-1} (\ldots (g_i + d_i) \ldots - e_{ij} \ldots) \otimes B_i^T \).

#### B. HAMILTON FUNCTION FOR DYNAMIC GRAPHICAL GAMES

According to the error dynamics (7) and the performance index (9), the Hamiltonian function of agent \( i \) can be defined as

\[
H_i(e_{ik}, \lambda_{ik+1}, u_{ik}, u_{-ik}) = U_i(e_{ik}, u_{ik}, u_{-ik}) + \lambda_{ik+1}^T f_i(e_{ik}, u_{ik}, u_{-ik})
\]

(17)

where, \( \lambda_{ik} \) is the Lagrangian multiplier vector. According to [24], the Lagrangian multiplier vector for optimal control can be written as

\[
\frac{\partial H_i(e_{ik}, \lambda_{ik+1}, u_{ik}, u_{-ik})}{\partial \lambda_{ik}} = A^T \lambda_{ik+1} + Q_{il} e_{ik} = \lambda_{ik}.
\]

(18)

So the optimal policy based on the Hamiltonian function is given by applying the stationarity condition \( \partial H_i / \partial u_{ik} = 0 \) such that

\[
u_{ik}^o = \arg \min_{u_{ik}} (H_i(e_{ik}, \lambda_{ik+1}, u_{ik}, u_{-ik})).
\]

(19)

Then we can obtain

\[
u_{ik}^o = (d_i + g_i) R_i^{-1} B_i^T \lambda_{ik+1}.
\]

(20)
C. Bellman Function Based on Q-Function

The Q-function of agent $i$ is defined as follows:

$$Q^i_0(\tilde{e}_{ik}, u_{ik}) = \frac{1}{2} \left( e_{ik}^T Q_0 e_{ik} + u_{ik}^T R_{ii} u_{ik} + \sum_{j \in N_i} u_{ij}^T R_{ij} u_{ij} \right) + V^\pi_i(\tilde{e}_{i(k+1)}) ,$$  \hspace{1cm} (19)

where, $Q_{ii} > 0 \in R^{n_i \times n_i}$, $R_{ii} > 0 \in R^{n_i \times n_i}$, $R_{ij} > 0 \in R^{n_i \times n_j}$ are the symmetric time-invariant weighting matrices. For the multi-agent graphical game optimization problem, the objective is to find the optimal value

$$J^*_i = \min_{u_{ik}} \sum_{k=0}^{\infty} U_i(\tilde{e}_{ik}, u_{ik}, u_{-ik}),$$  \hspace{1cm} (20)

where $u^*_{ik}$ denotes the optimal policies in the neighboring policies. Since policies $\pi_{jk}$, $(j \in N_i)$ are stationary admissible, there exist the best response Bellman equation. Note that $Q^\pi_i(\tilde{e}_{ik}, u_{ik})$ is defined as

$$Q^\pi_i(\tilde{e}_{ik}, u_{ik}) = V^\pi_i(\tilde{e}_{ik}).$$

So the best response Bellman function is defined as

$$Q^i_0(\tilde{e}_{ik}, u_{ik}) = U_i(\tilde{e}_{ik}, u_{ik}, \pi_{-ik}) + Q^\pi_i(\tilde{e}_{i(k+1)}, \pi_{i(k+1)}).$$ \hspace{1cm} (21)

with the initial condition $Q^i_0(0) = 0$, $\forall i$.

The difference of Q-function is defined as follows:

$$\Delta Q^i_0(\tilde{e}_{ik}, u_{ik}) = Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)}) - Q^i_0(\tilde{e}_{ik}, u_{ik}),$$ \hspace{1cm} (22)

and its gradient is

$$\nabla Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)}) = \frac{\partial Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)})}{\partial \tilde{e}_{i(k+1)}}.$$ \hspace{1cm} (23)

According to the Bellman function, when its gradient

$$\frac{\partial Q^i_0(\tilde{e}_{ik}, u_{ik})}{\partial u_{ik}} = R_{ii} \tilde{u}_{ik} + \frac{\partial Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)})}{\partial \tilde{e}_{i(k+1)}} = 0,$$ \hspace{1cm} (24)

we will have the optimal control policy of agent $i$ defined as follows:

$$\tilde{u}_{ik} = (g_i + d_i) R_{ii}^{-1} B_i^T \frac{\partial Q^i_0(\tilde{e}_{i(k+1)}, \pi_{i(k+1)})}{\partial \tilde{e}_{i(k+1)}} - R_{ii}^{-1} \left( \left[ e_i e_{i+1} \ldots \right] \otimes B_i^T \frac{\partial Q^i_0(\tilde{e}_{i(k+1)}, \pi_{i(k+1)})}{\partial e_{i(k+1)}} \right),$$

which is same as the best policy mentioned in (14), i.e.,

$$\pi_{ik} = \tilde{u}_{ik}^0 = M_i \nabla Q^i_0(\tilde{e}_{i(k+1)}, \pi_{i(k+1)}).$$

So the best response Bellman function based on the Q-function is defined as

$$Q^i_0(\tilde{e}_{ik}, \tilde{u}_{ik}) = \frac{1}{2} \left( e_{ik}^T Q_0 e_{ik} + \tilde{u}_{ik}^T R_{ii} \tilde{u}_{ik} + \sum_{j \in N_i} \tilde{u}_{ij}^T R_{ij} \tilde{u}_{ij} \right) + Q^i_0(\tilde{e}_{i(k+1)}, \tilde{u}_{i(k+1)}),$$ \hspace{1cm} (25)

which can be written as

$$Q^i_0(\tilde{e}_{ik}, \tilde{u}_{ik}) = Q^i_0(\tilde{e}_{i(k+1)}, \tilde{u}_{i(k+1)}) + \frac{1}{2} \left( e_{ik}^T Q_0 e_{ik} \right)$$

$$+ \frac{1}{2} \nabla Q^i_0(\tilde{e}_{i(k+1)}, \tilde{u}_{i(k+1)}) M_i^T R_{ii} M_i$$

$$+ \frac{1}{2} \sum_{j \in N_i} \nabla Q^i_0(\tilde{e}_{i(k+1)}, \tilde{u}_{i(k+1)}) M_j^T R_{ij} M_j$$

$$+ \nabla Q^i_0(\tilde{e}_{i(k+1)}, \tilde{u}_{i(k+1)}).$$ \hspace{1cm} (26)

D. Coupled Hamilton-Jacobi-Bellman Equations

According to [25], the Hamilton-Jacobi theory is used to relate the Hamiltonian function and the Bellman equation. The Discrete-Time Hamilton-Jacobi (DTHJ) equation is

$$\Delta V_i(\tilde{e}_{ik}) - \nabla V_i(\tilde{e}_{i(k+1)})^T \tilde{e}_{i(k+1)} + H_i(\tilde{e}_{ik}, \nabla V_i(\tilde{e}_{i(k+1)}), u_{ik}, u_{-ik}) = 0.$$ \hspace{1cm} (27)

Combining with the Q-function, we obtain the DTHJ equation as follows:

$$\Delta Q^i_0(\tilde{e}_{ik}, u_{ik}) - \nabla Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)})^T \tilde{e}_{i(k+1)} + H_i(\tilde{e}_{ik}, \nabla Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)}), u_{ik}, u_{-ik}) = 0.$$ \hspace{1cm} (28)

According to the Hamiltonian function, the Lagrangian multiplier vector for Q-function can be written as

$$\lambda_{i(k+1)} = \nabla Q^i_0(\tilde{e}_{i(k+1)}, \pi_{i(k+1)}).$$ \hspace{1cm} (27)

The following definition relates the Hamiltonian (15) along the optimal trajectories and (26).

Definition 1: [Discrete-Time Hamilton-Jacobi-Bellman Equation [25] a] Let $0 < Q^\pi_0(\tilde{e}_{ik}, u_{ik}^0) \in C^2$, $\forall i$, satisfy the Discrete-Time Hamilton-Jacobi-Bellman (DTHJB) equation:

$$H_i^T(\tilde{e}_{ik}, \nabla Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)}), u_{ik}, u_{-ik})$$

$$= \nabla Q^i_0(\tilde{e}_{i(k+1)}, u_{i(k+1)})^T \tilde{e}_{i(k+1)}$$

$$+ \frac{1}{2} \left( e_{ik}^T Q_0 e_{ik} + \tilde{u}_{ik}^T R_{ii} \tilde{u}_{ik} + \sum_{j \in N_i} \tilde{u}_{ij}^T R_{ij} \tilde{u}_{ij} \right) = 0.$$ \hspace{1cm} (28)

with initial condition $Q^i_0(0) = 0$, and the best policy is

$$u_{ik}^0 = (g_i + d_i) R_{ii}^{-1} B_i^T \lambda_{i(k+1)}.$$ \hspace{1cm} (29)

Then, $Q^{\pi_0}_i(\tilde{e}_{ik}, u_{ik}^0)$ satisfies (26).

b) Let $(A, B_i), \forall i$, be reachable for agent $i$. Let $0 < Q^{\pi_0}_i(\tilde{e}_{ik}, u_{ik}^0) \in C^2$, $\forall i$, satisfy (26). Then $Q^{\pi_0}_i(\tilde{e}_{ik}, u_{ik}^0)$ satisfies the DTHJ equation. The proof is given by Theorem 1 in [25].
IV. NASH SOLUTION FOR THE DYNAMIC GRAPHICAL GAME

When an agent’s policy achieves the optimal value of its expected return, all other agents also follow this policy [6], [25]. The objective of solving the Nash equilibrium of the dynamic graphical game is to solve (13), leading to (26).

A. NASH EQUILIBRIUM FOR THE GRAPHICAL GAMES

A Nash equilibrium solution for the game is given with respect to the actions of the other players \( u_i = \{ u_j | j \in N, j \neq i \} \) in the graph.

Definition 2: If all agents \( \forall i \in N \) satisfy

\[
J^*_i = J_i(u^*_i, u^*_j) \leq J_i(u_i, u^*_j),
\]

then an N-player game with N-tuple of optimal control policies \( \{u^*_1, u^*_2, \ldots, u^*_N\} \) is said to have a Nash equilibrium solution. So the Nash equilibrium of Q-function can be written as

\[
Q^\pi(i) = Q^\pi_i(\bar{e}_i, u^*_i) \leq Q^\pi_i(\bar{e}_i, u^*_k),
\]

where, \( Q_i \) is the local performance index of Q-function [6], [25].

B. STABILITY AND NASH SOLUTION FOR THE GRAPHICAL GAMES

A stable Nash solution for the dynamic graphical game is shown to be equivalent to the solution of (26). Definition 3: (Stability and Nash equilibrium solution) Let \( 0 < Q^\pi_i(\bar{e}_i, u^*_i) \in C^2, \forall i \), satisfy the Bellman optimality equation and all agents use the policies given by (26).

a) The error dynamics (7) is asymptotically stable and all agents synchronize to the leader’s dynamics (2).

b) The optimal performance index for agent \( i \) is \( J^*_ii = Q^\pi_i(\bar{e}_i, u^*_i) \).

\[ c) \text{All agents are in Nash equilibrium, with } J_i(u_i, u^*_j) \leq J_i(u_i, u^*_j) \text{ } [6], [25]. \]

The proof is given by Theorem 2 in [25].

C. BEST RESPONSE SOLUTION OF DYNAMIC GRAPHICAL GAMES

Consider that neighbor policies \( u_{-i} = \{ u_j : j \in N \} \) are fixed. The best response Bellman equation based on Q-function for agent \( i \) can be defined as

\[
Q_i(\bar{e}_i, u^*_i) = Q_i(\bar{e}_{i(k+1)}, u^*_{i(k+1)}) + \frac{1}{2} X^TM^T \bar{R}_iM_iX + \frac{1}{2} \left( \bar{e}_i^T Q_i(\bar{e}_i) + \sum_{j \in N_i} u^T_{ij}R_{ij}u_{ij} \right)
\]

with the initial condition \( Q^0_i(0, 0) = 0, X = \nabla Q^0_i(\bar{e}_{i(k+1)}, u^*_{i(k+1)}), M_i = R_{ii}^{-1} \left( \begin{array}{c} g_i + d_i \\ \vdots \\ -e_j \end{array} \right) \otimes B_i \),

and \( u_{ik} \) is given by (17).

The best response Hamilton-Jacobi (HJ) equation is defined as

\[
\bar{H}^1_i(\bar{e}_i) = \nabla Q^\pi_i(\bar{e}_i, u^*_i, u^*_{-i}(-\bar{e}_i)) = \nabla Q^\pi_i(\bar{e}_{i(k+1)}, u_{ik+1}) + u^T_{ik}R_{ik}u_{ik} + \sum_{j \in N_i} u^T_{ij}R_{ij}u_{ij} \]

\[ + \frac{1}{2} \left( u^T_{ik}Q_{ik} + u^T_{ik}R_{ik}u_{ik} + \sum_{j \in N_i} u^T_{ij}R_{ij}u_{ij} \right) = 0, \]

with the initial condition \( Q^\pi_i(0, 0) = 0, \) and \( u_{ik} = u^*_i \) is given by (29).

The following lemma shows the relation between (32) and (33). Lemma 1: [6] Let \( 0 < Q^\pi_i(\bar{e}_i, u^*_i) \in C^2, \forall i, \) satisfy (33) with the initial condition given by \( Q^\pi_i(0, 0) = 0, \) and the optimal control policy is given by (29). Then \( Q^\pi_i(\bar{e}_i, u^*_i) \) satisfies (32).

b) Let \( A, B_i, \forall i, \) be reachable for agent \( i, \) and \( 0 < Q^\pi_i(\bar{e}_i, u^*_i) \in C^2, \forall i, \) satisfy (32). Then \( Q^\pi_i(\bar{e}_i, u^*_i) \) satisfies (33).

c) All agents are in Nash equilibrium.

The proof of Lemma 1 is given in Appendix.

Theorem 1: (Best Response Solution) According to [25], [26], for fixed neighbor policies \( u_{-i} = \{ u_j : j \in N \} \), assume there exist an admissible policy \( u_i \). Let \( Q^\pi_i(\bar{e}_i, u^*_i) \) satisfying (32) or (33), and agent \( i \) uses control policy (18). Then:

a) The error dynamics (7) is asymptotically stable and all agents synchronize to the leader’s dynamics (2).

b) The optimal performance index for agent \( i \) is \( J^*_ii = Q^\pi_i(\bar{e}_i, u^*_i) \).

c) All agents are in Nash equilibrium.

The proof of Theorem 1 is given in Appendix.

V. THE PROPOSED VALUE ITERATION ALGORITHM

This section proposes a value iteration algorithm to solve discrete-time dynamic graphical games in real time by using a cooperative agent RL algorithm. The single agent RL algorithm is extended to solve multi-agent dynamic graphical games, with a simplified writing of the value function based on Q-function, which consists of the control input and the state of the agent as well as the state of its neighbors, and is defined as follows:

\[
\bar{Q}_i = \frac{1}{2} \left[ \bar{e}_i^T u^T_{ik} \right] \bar{H}_i \left[ \bar{e}_i u_{ik} \right],
\]

and

\[
\bar{H}_i = \left[ \bar{H}_i(\bar{e}_i, \bar{e}_j, \bar{e}_k) \bar{H}_i(\bar{e}_i, \bar{e}_j) \ldots \bar{H}_i(\bar{e}_i, \bar{e}_k) \bar{H}_i(\bar{e}_i, u_{ik}) \right] = \left[ \bar{H}_i(\bar{e}_i, \bar{e}_j, \bar{e}_k) \bar{H}_i(\bar{e}_i, \bar{e}_j) \ldots \bar{H}_i(\bar{e}_i, \bar{e}_k) \bar{H}_i(\bar{e}_i, u_{ik}) \right],
\]

where, for agent \( i, \) \( \bar{H}_i(\bar{u}_{ik}) \) is the solution matrix. \( \bar{H}_i(u_{ik}, e_{ik}) \) is a sub-block of matrix \( \bar{H}_i \). The diagonal element of \( \bar{H}_i \) is the agent’s state vector. \( \bar{H}_i(u_{ik}, u_{ik}) \) and \( \bar{H}_i(u_{ik}, e_{ik}) \) are the weighting matrices of the agent’s policy,
the neighbor agents and the policies of the neighbor agents, respectively.

The following gives the optimal control policy, which solves out the minimum by letting the partial derivative be equal to 0, i.e.,

$$\frac{\partial Q}{\partial u_{ik}} = 0 \Rightarrow 2\tilde{H}_i (u_{ik} e_{ik}) + 2\tilde{H}_i (u_{ik} e_{ik}) e_{ik} = 0.$$  

We obtain

$$u_{ik} = -\tilde{H}^{-1}_i (\nabla_{u_{ik}} \tilde{H}_i (u_{ik} e_{ik}) + \sum_{j \in N_i} \tilde{H}^{-1}_j (\nabla_{u_{ik}} \tilde{H}_j (u_{ik} e_{ik}))) e_{jk}.$$  

(36)

Algorithm 1 is proposed to solve (66) for the multi-agent system.

---

**Algorithm 1 The Proposed Value Iteration Algorithm**

1. Begin from a random initial policy $u^0_{ik}$ and value $Q^0_i (e_{ik}, u^0_{ik})$.
2. Start action with the policy $u^0_{ik}$ and value $Q^0_i (e_{ik}, u^0_{ik})$.
3. Obtain local error state vector $\tilde{e}_{ik+1}$ at the next moment with (7), and solve for $Q^{[l+1]}_i$ by using

$$Q^{[l+1]}_i (\tilde{e}_{ik}, u^l_{ik}) = Q^{[l]}_i (\tilde{e}_{ik+1}, u^l_{ik+1}) + U_i (\tilde{e}_{ik}, u^l_{ik}, u^l_{i-ik}).$$  

(37)

4. Update policy $u^{[l+1]}_{ik}$ by using

$$u^{[l+1]}_{ik} = -\tilde{H}^{-1}_i (\nabla_{u_{ik}} \tilde{H}_i (u_{ik} e_{ik}) + \sum_{j \in N_i} \tilde{H}^{-1}_j (\nabla_{u_{ik}} \tilde{H}_j (u_{ik} e_{ik}))) e_{jk}, \forall i.$$  

(38)

5. Repeat steps 3 and 4, and stop until reaching $\|Q^{[l+1]}_i - Q^{[l]}_i\|_{\infty}$ consensus.

---

**Remark 2:** The algorithm does not require the knowledge of any of the agents’ dynamics in the system (7).

**Remark 3:** With (34) and (35), (36) enables Algorithm 1 to solve dynamic graphical games for both the directed and undirected graph topologies [25].

**Remark 4:** The main differences between the algorithm proposed here and that in [6] are the value function and the way to update policies. The value function of the algorithm in [6] is defined by

$$H_i (e_{ik}, \lambda_{ik+1}, u_{ik}, u_{-ik}) = \lambda^T_{ik+1} (A e_{ik} - (d_i + g_i) B_i u_{ik} + \sum_{j \in N_i} e_{jk} B_j u_{jk}).$$

$$+ \frac{1}{2} \left( e^T_{ik} Q_{ik} e_{ik} + u^T_{ik} R_{ik} u_{ik} + \sum_{j \in N_i} u^T_{ik} R_{ik} u_{jk} \right),$$

with an updating policy is defined by

$$u^{l+1}_{ik} = (d_i + g_i) R_{ik} u_{ik} + \sum_{j \in N_i} u^T_{ik} R_{ik} u_{jk},$$  

(39)

which requires system dynamics $B$ that is the parameter of the model, but not for our proposed algorithm. Since the updating policy in [1] also requires system dynamics, the algorithm proposed in [1] is model-based.

**Theorem 2:** (Value iteration algorithm convergence) Let all agents execute Algorithm 1 at the same time. Assume that all initial policies are acceptable. Suppose that $\tilde{\sigma} R^{-1}_{ik} \forall i$, are all small, and the initial condition is $Q^0_i (e_{ik}) = 0$.

a) First let agent $i$ perform Algorithm 1 while its neighbor agents’ policies are fixed, and the learning rate is clear and stable. Assume there exists an admissible policy $u_{ik}$. Let agent $i$ perform Algorithm 1. Then the solution sequence converges to (28).

b) Assume there exists admissible policies. Let all agents update their policies by using Algorithm 1. Then the solution sequence $\{Q^0_i \}_{l=0}^{\infty}$ converges to (28).

c) Since Algorithm 1 is a value iteration algorithm, its convergence speed is faster than that of the policy iteration.

**Proof:** a) When the neighbor policies are fixed, the difference of Q-function (37) yields for $\forall i, l$,

$$Q^{[l]}_i (\tilde{e}_{ik}, u^l_{ik}) - \tilde{Q}^0_i (\tilde{e}_{ik+1}, u^l_{ik+1}) = \frac{1}{2} \left( e^T_{ik} Q_{ik} e_{ik} + u^T_{ik} R_{ik} u_{ik} + \sum_{j \in N_i} u^T_{ik} R_{ik} u_{jk} \right).$$  

(40)

We obtain

$$\tilde{Q}^{[l+1]}_i (\tilde{e}_{ik}, u^l_{ik}) > \tilde{Q}^0_i (\tilde{e}_{ik+1}, u^l_{ik+1}).$$

Rearranging (40) yields

$$\tilde{Q}^{[l+1]}_i (\tilde{e}_{ik}, u^l_{ik} | \pi) - \tilde{Q}^0_i (\tilde{e}_{ik}, u^l_{ik} | \pi) = \tilde{Q}^0_i (\tilde{e}_{ik+1}, u^l_{ik+1} | \pi) - \tilde{Q}^{[l]}_i (\tilde{e}_{ik+1}, u^l_{ik+1} | \pi) = \tilde{Q}^{[l-1]}_i (\tilde{e}_{ik+2}, u^l_{ik} | \pi) - \tilde{Q}^{[l-2]}_i (\tilde{e}_{ik+2}, u^l_{ik} | \pi) \ldots = \tilde{Q}^{[l]}_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi) - \tilde{Q}^0_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi).$$  

(41)

We have

$$\tilde{Q}^{[l+1]}_i (\tilde{e}_{ik}, u^l_{ik} | \pi) = \tilde{Q}^0_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi) = \tilde{Q}^{[l]}_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi) = \tilde{Q}^{[l-1]}_i (\tilde{e}_{ik+l-1}, u^l_{ik+l-1} | \pi) + \tilde{Q}^{[l]}_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi) = \tilde{Q}^{[l]}_i (\tilde{e}_{ik+l}, u^l_{ik} | \pi) + \ldots \tilde{Q}^{[l]}_i (\tilde{e}_{ik+l-1}, u^l_{ik+l-1} | \pi).$$
For all policies \( u'_{ik} \), they have a monotonically increasing sequence of value functions \( \tilde{Q}_{ik} \), \( \forall i \), such that

\[
0 < \ldots \tilde{Q}_{i}^{0} < \ldots < \tilde{Q}_{i}^{*} < \tilde{Q}_{i}^{+1}, \forall i, l.
\]

From (19), we can obtain that \( \tilde{Q}_{ik}^{+1} \), \( \forall i \), is bounded by \( \tilde{Q}_{i}^{+1} \), which means that \( \tilde{Q}_{ik}^{+1} \), \( \forall i \), has an upper bound. So Algorithm 1 converges to (32) as follows:

\[
\tilde{Q}_{i}^{*} (\tilde{e}_{ik}, u_{ik}) = \frac{1}{2} \left( \tilde{e}_{ik}^{T} Q_{i}^{*} \tilde{e}_{ik} + \tilde{u}_{ik}^{T} R_{i} \tilde{u}_{ik} + \sum_{j \in N_{i}} L_{ik}^{T} R_{ij} \tilde{u}_{jk} \right) + \tilde{Q}_{i}^{*} \left( \tilde{e}_{(ik+1)}, \tilde{u}_{(ik+1)}^{*} \right),
\]

where

\[
\tilde{u}_{ik}^{*} = M_{i} \nabla Q_{i}^{*} (\tilde{e}_{i(k+1)}, u_{i(k+1)}).
\]

So (46) satisfies Theorem 1 in Section IV. 

b) For the second case, we have

\[
\tilde{Q}_{i}^{+1} (\tilde{e}_{ik}) - \tilde{Q}_{i}^{*} (\tilde{e}_{ik}) = \tilde{Q}_{i}^{l} (\tilde{e}_{i(k+1)}) - \tilde{Q}_{i}^{l-1} (\tilde{e}_{i(k+1)}) = \tilde{Q}_{i}^{l-1} (\tilde{e}_{i(k+2)}) - \tilde{Q}_{i}^{l-2} (\tilde{e}_{i(k+2)}) \ldots = \tilde{Q}_{i}^{l-1} (\tilde{e}_{i(k+l)}) - \tilde{Q}_{i}^{l} (\tilde{e}_{i(k+l)}).
\]

After manipulation, it can be rewritten as

\[
\tilde{Q}_{i}^{+1} (\tilde{e}_{ik}) = \tilde{Q}_{i}^{l} (\tilde{e}_{i(k+l)}) + \tilde{Q}_{i}^{l-1} (\tilde{e}_{i(k+l-1)}) + \ldots + \tilde{Q}_{i}^{2} (\tilde{e}_{i(k+2)}) + \tilde{Q}_{i}^{1} (\tilde{e}_{i(k+1)}).
\]

We can also obtain \( \tilde{Q}_{i}^{+1} (\tilde{e}_{ik}) \leq \sum_{c=0}^{\infty} \tilde{Q}_{i}^{l} (\tilde{e}_{i(k+c)}). \) Then, for all policies \( u_{ik}^{l} \), it has a monotonically increasing sequence of value functions \( \tilde{Q}_{ik}^{+1} \), \( \forall i \), such that

\[
0 < \ldots \tilde{Q}_{i}^{0} < \ldots < \tilde{Q}_{i}^{*} < \tilde{Q}_{i}^{+1}, \forall i, l.
\]

From (19), we can obtain that \( \tilde{Q}_{ik}^{+1} \), \( \forall i \), is bounded by \( \tilde{Q}_{i}^{+1} \) with an upper bound that Algorithm 1 converges to the best response Bellman function (25), with

\[
\tilde{u}_{ik}^{*} = M_{i} \nabla Q_{i}^{*} (\tilde{e}_{i(k+1)}, u_{i(k+1)}).
\]

C) With the policy iteration algorithm in [25], each iteration will improve the policy and calculate the value. However, each iteration will be calculated from the start state to the end state rather than the previous value as follows:

\[
\tilde{Q}_{i}^{l} (\tilde{e}_{ik}, u_{ik}) = U_{i} (\tilde{e}_{ik}, u_{ik}, u_{ik}^{l-1}) + \tilde{Q}_{i}^{l-1} (\tilde{e}_{i(k+1)}, u_{i(k+1)}^{l-1}).
\]

With our proposed algorithm, the calculation of the value is based on the value from the last iteration. So it converges faster than the policy iteration.

VI. GRAPHICAL GAME SOLUTION BASED ON Q-FUNCTION BY ACTOR-CRITIC LEARNING

This section develops an actor-critic network to implement the proposed Algorithm 1 for solving the dynamic graphical games in real-time. Each agent has its own critic network to perform the value update, and the actor network to perform the policy update. The actor-critic network structures depend only on the local information of agents.

A. ACTOR-CRITIC NETWORKS

The value function for agent \( i, Q_{i} (\tilde{e}_{ik}, u_{ik}) \), is approximated by the critic network \( \hat{Q}_{i} (\tilde{W}_{ic}) \). The control policy \( \tilde{u}_{ik} (\tilde{W}_{ia}) \) is approximated by an actor network such that

\[
\hat{Q}_{i} (\tilde{W}_{ic}) = \frac{1}{2} \tilde{T}_{ic}^{T} \tilde{W}_{ic} L_{ic},
\]

\[
\tilde{u}_{ik} (\tilde{W}_{ia}) = \tilde{T}_{ia}^{T} \tilde{W}_{ia} L_{ia}.
\]

where, \( \tilde{W}_{ic} \in R^{n_{i} \times (n_{i} + m_{i})}, \forall i \), are the weighting matrices of the approximated structures \( \hat{Q}_{i} (\tilde{W}_{ic}) \), \( \tilde{W}_{ia} \in R^{m_{i} \times m_{i}} \), are actor weights. \( L_{ic} = [e_{ic}^{T} \ldots e_{ic}^{T}]^{T} \) consists of state \( e_{ic} \) and the state of its neighbours \( e_{ic} \). Set \( \tilde{Q}_{ia} (\tilde{e}_{ia}, \tilde{u}_{ia}) \) to be the approximation error of the actor network so that

\[
\tilde{e}_{ia} (\tilde{e}_{i}, \tilde{u}_{ia}) = \tilde{u}_{ia} (\tilde{W}_{ia}) - \tilde{u}_{ia} = \tilde{T}_{ia}^{T} \tilde{W}_{ia} L_{ia} - \tilde{u}_{ia}.
\]

The update of control policies with the critic structure is given by (36), and

\[
\tilde{u}_{ik} = -\tilde{W}_{ic}^{-1} \tilde{W}_{ic} (\tilde{u}_{ia}, \tilde{u}_{ia}) \tilde{e}_{ik} - \sum_{j \in N_{i}} \tilde{W}_{ic}^{-1} \tilde{W}_{ic} (\tilde{u}_{ia}, \tilde{u}_{ia}) \tilde{e}_{ik},
\]

where, \( \tilde{W}_{ic} (\tilde{u}_{ia}, \tilde{u}_{ia}) \) represents the block matrix defined by the positions of control approximation and the state of agent \( i \).

The square approximation error is

\[
err_{actor} = \frac{1}{2} \left( \tilde{Q}_{ia} (\tilde{e}_{ia}, \tilde{u}_{ia}) \right)^{T} \tilde{Q}_{ia} (\tilde{e}_{ia}, \til{u}_{ia}).
\]

The update rules for the actor weights are given by

\[
\tilde{W}_{ia}^{(l+1)T} = \tilde{W}_{ia}^{(l)T} - \mu_{ia} \left( (\tilde{W}_{ia}^{(l)T} \tilde{T}_{ia}^{T} - \tilde{u}_{ia}) (L_{ia})^{T} \right),
\]

where, \( 0 < \mu_{ia} < 1 \) is the learning rate of the actor network.
Set $\tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k})$ to be the target value of the neural critic network structure at step $l$ such that

$$
\tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k}) = \frac{1}{2} \left( e_{i,k}^T Q_{\tilde{e}_{i,k}} + \tilde{u}_{i,k}^T R_{i,k} + \sum_{j \in N_i} \tilde{u}_{j,k}^T R_{i,k} \tilde{u}_{j,k} \right) + \tilde{Q}_i(\tilde{e}_{i,k+1}, \tilde{a}_{i,k+1}).
$$

(55)

The critic network approximation error at step $l$ is given by

$$
\xi_{i,k} = \tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k}) - \hat{Q}_i(\tilde{W}_i).\quad (56)
$$

The square approximation error for the critic network is defined as

$$
err_{critic} = \frac{1}{2} (\xi_{i,k})^T (\xi_{i,k}) = \frac{1}{2} \| \tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k}) - L_{ik} \tilde{W}_i^T L_{ik} \|^2.\quad (57)
$$

The update rules for the critic network weights are given by

$$
\tilde{W}_{i,k}^{(l+1)} = \tilde{W}_{i,k}^{(l)} - \mu_{ic} \left( \tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k}) - L_{ik} \tilde{W}_i^T L_{ik} \right) L_{ik} L_{ik}^T.
$$

**Algorithm 2** The Proposed Value Iteration Algorithm Based on Actor-Critic Networks

1. Initialize value actor weights $\tilde{W}_i^0$ and critic weights $\tilde{W}_{i,k}^0$.

2. repeat

   (l is the iteration index)

3. Begin from initial state $\tilde{e}_{i,0}$ on the system trajectory.

4. Use (51) to calculate $\tilde{u}_{i,k}$ in Algorithm 1.

5. Use (7) to calculate dynamics $e_{i,k+1}$.

6. Use (53) and critic weights $\tilde{W}_i$ to calculate $\tilde{u}_{i,k}$.

7. Use (51) to calculate $\tilde{Q}_i^{(l)}(\tilde{e}_{i,k}, \tilde{a}_{i,k})$, $\forall i$, in Algorithm 1.

8. Critic update rule:

   $$
   \tilde{W}_{i,k}^{(l+1)} = \tilde{W}_{i,k}^{(l)} - \mu_{ic} \left( \tilde{Q}_i(\tilde{e}_{i,k}, \tilde{a}_{i,k}) - L_{ik} \tilde{W}_i^T L_{ik} \right) L_{ik} L_{ik}^T
   $$

9. Actor update rule:

   $$
   \tilde{W}_i^{(l+1)} = \tilde{W}_i^{(l)} - \mu_{ia} \left( \tilde{Q}_i^{(l+1)}(\tilde{e}_{i,k}, \tilde{a}_{i,k}) - L_{ik} \tilde{W}_i^T L_{ik} \right),\quad \forall i
   $$

10. until reaching the consensus of $\| \tilde{Q}_i^{l+1} - \tilde{Q}_i^l \|$.  

**C. SIMULATION STUDIES**

In this section, we use simulation to verify the effectiveness of the algorithm proposed in this paper. The graph of the multi-agent is shown in Fig.1, where, Agent 0 is the leader and the others are followers.

The state and input of the agent system, which follows (1), are given by

$$
A = \begin{pmatrix} 0.995 & -0.0993 \\ -0.0993 & 0.995 \end{pmatrix},
$$

$$
B_1 = \begin{pmatrix} 0.2 \\ 0.1 \end{pmatrix},
$$

$$
B_2 = \begin{pmatrix} 0.2047 \\ 0.08984 \end{pmatrix},
$$

$$
B_3 = \begin{pmatrix} 0.2097 \\ 0.1897 \end{pmatrix}.
$$

Since the proposed scheme is model-free, $A, B_1, B_2$ and $B_3$ are just used to calculate the tracking error dynamics.

Pinning gains are as follows:

$$
e_{12} = 0.8, e_{21} = 0.7, e_{23} = 0.6, e_{31} = 0.8,\quad \text{and the others} = 0.
$$

Performance index weighting matrices are set as follows:

$$
Q_{11} = Q_{22} = Q_{33} = I_{2 \times 2},
$$

$$
R_{11} = R_{22} = R_{33} = 1,
$$

$$
R_{13} = R_{32} = 0,
$$

$$
R_{12} = R_{21} = R_{23} = R_{31} = 1.
$$

The initial policies, value actor weights and critic weights are given by

$$
u_1 = 0.1669, u_2 = 0.4054, u_3 = -0.1905,
$$

$$
W_{a1} = \begin{bmatrix} 0.1493 & 0.2575 & 0.8407 & 0.2543 \end{bmatrix},
$$

$$
W_{a2} = \begin{bmatrix} 0.8143 & 0.2435 & 0.9293 & 0.3500 \end{bmatrix},
$$

$$
W_{a3} = \begin{bmatrix} 0.1966 & 0.2511 \end{bmatrix},
$$

$$
W_{c1} = \begin{bmatrix} 0.0008 & 0.0053 & 0.0093 & 0.0057 \end{bmatrix},
$$

$$
W_{c2} = \begin{bmatrix} 0.0005 & 0.0078 & 0.0013 & 0.0047 \end{bmatrix},
$$

$$
W_{c3} = \begin{bmatrix} 0.0001 & 0.0016 & 0.0031 & 0.0017 \end{bmatrix},
$$

$$
W_{c4} = \begin{bmatrix} 0.0034 & 0.0079 & 0.0053 & 0.0060 \end{bmatrix},
$$

$$
W_{c5} = \begin{bmatrix} 0.0026 & 0.0069 \end{bmatrix}.
$$

The learning rates are $\mu_{ic} = 0.1$, and $\mu_{ia} = 0.1$, $\forall i$. 

**FIGURE 1.** Graphical example.
Fig. 2 shows the critic of agent 2 under the initial data, which has 2 neighbors with agent 2. The state vector of each agent here has two state components. Because the state vector matrix of each agent is two-dimensional, the state vector of each agent has two rows as state components. So the critic network for agent 2 is a $2 \times 4$ matrix, and the changes of 8 curves in Fig. 2 refers to the changes of 8 numbers in the matrix. Every 4 curves refer to an agent’s state, and we can see that some curves reach to straight state at the same time, such as $w_{c21}$, $w_{c22}$, $w_{c25}$ and $w_{c26}$.

Fig. 2(b) shows the actor weights of agent 2 under the initial data. Since the policy is a number rather than a matrix, and the state vector of each agent has two state components, the matrix of agent 2 is a $1 \times 4$ matrix. Similarly, every two curves refer to an agent’s policy state.

Fig. 3 shows the tracking error dynamic transformation of all the follower agents. By comparing the number of iterations that reach convergence, it can be concluded that the number of iterations required by the proposed algorithm is almost one-tenth of that of Fig. 3(b), so the convergence speed in Fig. 3(a) is much faster than that in Fig. 3(b) with about 10 times.

According to Figs. 2(a), 2(b), and 3(b), when the neighborhood tracking error comes to 0, the critic and actor weights reach to a steady state, and the Algorithm 1 reaches to convergence.

Fig. 4 shows the dynamics of all the follower agents. We can see that they finally synchronize to the leader while reaching their optimality. It is obvious that the number of iterations for the agent with the policy iteration to reach the state synchronization is more than that with the value iteration.

Fig. 5 compares the tracking error dynamic transformation results given by the proposed algorithm and the model-based algorithm proposed in [6].

Fig. 6 compares the tracking error dynamic transformation results given by the proposed algorithm and model-based policy iteration algorithm proposed in [1].

Therefore, according to Fig. 5 and Fig. 6, although the complexity of our model-free algorithm is relatively low, with no requirement of the knowledge of system dynamics, our algorithm can also make the tracking error dynamics quickly comes to 0.
FIGURE 4. Comparison of agent’s dynamic.

FIGURE 5. Comparison of tracking error dynamics.

FIGURE 6. Comparison of tracking error dynamics.

Fig. 7 compares the error dynamic transformation results under normal conditions and when the value function has a small calculation error at step 100. We can obtain that although the error dynamic transformation is different, the final result still follows Nash equilibrium after some iterations of policy updates.
VII. CONCLUSION

In this paper, we improve the convergence speed of solving the Nash equilibrium of multi-agent systems by replacing the policy iteration algorithm proposed in [25] with the proposed value iteration. The rationality and convergence of this algorithm are proved theoretically. An actor-critic network structure is used to implement this algorithm. The simulation results show that the convergence speed of the proposed value iteration algorithm is much faster than that of the policy iteration algorithm proposed in [25].

More research works are still necessary to further improve the proposal. First, there are many confrontations and cooperation between multiple agents, which are not taken into account by the proposal. Second, in unknown environments, deep learning and reinforcement learning algorithms need to be combined to improve the suitability of the proposal. Third, it is interesting to investigate how to apply the proposed algorithm in dynamic formation systems.

APPENDIX

THE PROOF OF LEMMA 1 AND THEOREM 1

A. THE PROOF OF LEMMA 1

Proof: [The proof of Lemma 1] a) If \( Q^*_i(\bar{e}_{ik}, u^*_{ik}) \) satisfies the Discrete-Time Hamilton-Jacobi (DTHJ) equation, then

\[
H^*_i(\bar{e}_{ik}, \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1)), u^*_{ik}, u^-_{ik}) = 0.
\]

Since

\[
\nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T = \frac{\partial H_i(\bar{e}_{ik}, \tilde{\lambda}_{ik}(k+1), u^*_{ik}, u^-_{ik})}{\partial \bar{e}_{ik}}
\]

\[
= A^T \tilde{\lambda}_{ik}(k+1) + Q_i(\bar{e}_{ik}),
\]

according to the DTHJ equation, we can obtain

\[
\nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T \tilde{e}_{ik}(k+1) = \lambda^T_{ik+1} \bar{e}_{ik}(k+1)
\]

\[
= -\frac{1}{2} \left( \varepsilon_{ik}^T Q_i(\varepsilon_{ik} + u_{ik}(k) R_{ij} u_{ik} + \sum_{j \in N_i} u_{ik}^T R_{ij} u_{jk} \right)
\]

\[
= \Delta Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1)).
\]

Therefore, \( Q^*_i(\bar{e}_{ik}, u^*_{ik}) \) satisfies (25) or (26).

b) The best response Hamiltonian with arbitrary smooth value \( Q^*_i(\bar{e}_{ik}, u^*_{ik}) \) for arbitrary control policy \( u_{ik} \) is given by

\[
H^*_i(\bar{e}_{ik}, \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1)), u^*_{ik}, u^-_{ik})
\]

\[
= \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T \tilde{e}_{ik}(k+1) + U_i(\bar{e}_{ik}, u_{ik})
\]

\[
= H^*_i(\bar{e}_{ik}, \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1)), u^*_{ik}, u^-_{ik})
\]

\[
+ \frac{1}{2} (u_{ik} - u^*_{ik})^T R_{ij} (u_{ik} - u^*_{ik})
\]

\[
(59)
\]

Now set \( Q^*_i(\bar{e}_{ik}, u_{ik}) \) to satisfy (25) such that

\[
Q^*_i(\bar{e}_{ik}, u_{ik})
\]

\[
= U_i(\bar{e}_{ik}, u_{ik}) + Q^*_i(\bar{e}_{ik}(k+1), \pi_{ik}(k+1))
\]

\[
= U_i(\bar{e}_{ik}, u_{ik}) + Q^*_i(\bar{e}_{ik}(k+1), \pi_{ik}(k+1))
\]

\[
+ \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T \tilde{e}_{ik}(k+1)
\]

\[
- \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T \bar{e}_{ik}(k+1)
\]

\[
(60)
\]

By applying the Bellman optimality principle, \( Q^*_i(\bar{e}_{ik}, u_{ik}) \) has to satisfy the following equation:

\[
Q^*_i(\bar{e}_{ik}, u_{ik}) = \min_{u_{ik}} \left( Q^*_i(\bar{e}_{ik}(k+1), \pi_{ik}(k+1)) \right)
\]

\[
+ \frac{1}{2} (u_{ik} - u^*_{ik})^T R_{ij} (u_{ik} - u^*_{ik})
\]

\[
- \nabla Q^*_i(\bar{e}_{ik}(k+1), u_{ik}(k+1))^T \tilde{e}_{ik}(k+1)
\]

\[
(61)
\]

By applying the stationarity condition of the Bellman optimality principle as follows:

\[
\partial Q^*_i(\bar{e}_{ik}, u_{ik}) / \partial u_{ik} = 0,
\]

\[
(62)
\]
we have to solve the following equation for the control policy $u_{ik}^n$, i.e.,

$$u_{ik}^n - u_{ik}^* = (g_i + d_i)R_{ii}^{-1}B_i^T(A^T)^T(\nabla Q_i^* (\tilde{e}_{ik}, u_{ik}^*)) - \nabla Q_i^* (\tilde{e}_{ik+1}, u_{ik+1}^*))$$

(62)

Since

$$\frac{\partial H}{\partial u_{ik}} = \frac{\partial Q_i}{\partial u_{ik}} = R_{ii},$$

with the optimal control policy $u_{ik}^n = u_{ik}^*$, $\forall k$, (62) and the Lagrangian multiplier vector (16) yield

$$(g_i + d_i)R_{ii}^{-1}B_i^T(\nabla Q_i^* (\tilde{e}_{ik}, u_{ik}^*)) - \nabla Q_i^* (\tilde{e}_{ik+1}, u_{ik+1}^*)) = 0.$$ 

Furthermore, since $Q_i^*(0, 0) = Q_i^*(0, 0) = 0$, we can obtain

$$Q_i^* (\tilde{e}_{ik}, u_{ik}^*) = Q_i^* (\tilde{e}_{ik}, u_{ik}^*).$$

(63)

B. THE PROOF OF THEOREM 1

Proof: [The proof of Theorem 1] a) Since $Q_i^* (\tilde{e}_{ik}, u_{ik}^*)$ satisfies (32), we can have

$$Q_i^* (\tilde{e}_{ik}, u_{ik}^*) - Q_i^* (\tilde{e}_{ik+1}, u_{ik+1}^*)$$

$$= \frac{1}{2} \left( \tilde{e}_{ik}^T Q_i^* u_{ik}^* + u_{ik}^T R_i u_{ik}^* + \sum_{j \in N_i} u_{ik}^T R_{ij} u_{jk}^* \right).$$

(64)

Therefore, $Q_i^* (\tilde{e}_{ik}, u_{ik}^*)$ serves as a Lyapunov function for the error system (7), which is asymptotically stable. According to (6), all agents synchronize to the target nodes’ dynamics.

b) First complete the squares on the Hamiltonian of (15) for arbitrary control policies, then use Definition 1 and Discrete-Time Hamilton-Jacobi-Bellman Equation (DTHJB) (28) to yield

$$H_i^* (e_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}, \pi_{ik})$$

$$= H_i^* (\tilde{e}_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}, \pi_{ik} - u_{ik})$$

$$+ \frac{1}{2} (u_{ik} - u_{ik}^*)^T R_{ii} (u_{ik} - u_{ik}^*)$$

$$+ \sum_{j \in N_i} u_{ik}^T R_{ij} (\pi_{jk} - u_{jk}^*)$$

$$+ \frac{1}{2} \sum_{j \in N_i} (\pi_{jk} - u_{jk}^*)^T R_{ij} (\pi_{jk} - u_{jk}^*)$$

$$+ \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1})^T \tilde{g}_i B_i (\bar{u}_{ik} - u_{ik}^*)$$

$$+ \nabla Q_i^* (\bar{e}_{ik+1}, \pi_{ik})^T \sum_{j \in N_i} e_{ij} B_j (\pi_{jk} - u_{jk}^*),$$

where

$$u_{ik}^* = (d_i + g_i)R_{ii}^{-1}B_i^T(\nabla Q_i^* (\bar{e}_{ik+1}, u_{ik+1}^*)), \forall i.$$ 

Then $Q_i^* (\tilde{e}_{ik}, u_{ik}), \forall i$, satisfies (26).

Using the result from part a) to yield $Q_i^* (\tilde{e}_{i} (\infty), u_i (\infty)) \rightarrow 0$, we can have the performance index of an agent at time index $l$ as follows:

$$J_i^l = Q_i^* (\tilde{e}_{i} (\infty), u_i (\infty)) + \sum_{k=l}^{\infty} (U_{il} (e_{ik}, u_{ik}, \pi_{ik})),$$

(65)

which can be written as

$$J_i^l = Q_i^* (\tilde{e}_{il}, u_{il}^*)$$

$$+ \sum_{k=l}^{\infty} (U_{il} (e_{ik}, u_{ik}, \pi_{ik}) - U_{il}^* (\bar{e}_{ik}, \bar{u}_{ik}, \bar{u}_{ik}^*)).$$

(66)

The best response Hamiltonian with arbitrary control input $u_{ik}$ is given by

$$H_i^* (e_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}, \pi_{ik})$$

$$= \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1})^T \tilde{e}_{ik+1} |_{u_{ik}=u_{ik}^*}$$

$$+ U_{il}^* (\bar{e}_{ik}, \bar{u}_{ik}, u_{ik}^*).$$

(67)

The best response (HJ) equation for agent $i$ is given by

$$H_i^* (e_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}^*, u_{ik})$$

$$= \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1})^T \tilde{e}_{ik+1} |_{u_{ik}=u_{ik}^*}$$

$$+ U_{il}^* (\bar{e}_{ik}, \bar{u}_{ik}, u_{ik}^*) = 0.$$ 

(68)

We can obtain

$$U_i (e_{ik}, u_{ik}, u_{ik}), U_i^* (e_{ik}, u_{ik}, u_{ik}) - U_i^* (e_{ik}, u_{ik}, u_{ik})$$

$$= H_i^* (e_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}, \pi_{ik})$$

$$= H_i^* (e_{ik}, \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1}), u_{ik}^*, u_{ik})$$

$$- \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1})^T \tilde{e}_{ik+1} |_{u_{ik}=u_{ik}^*}$$

$$- \nabla Q_i^* (\tilde{e}_{ik+1}, \pi_{ik+1})^T \tilde{e}_{ik+1} |_{u_{ik}=u_{ik}^*}.$$ 

(69)

Finally we have

$$U_i (e_{ik}, u_{ik}, u_{ik}) - U_i^* (e_{ik}, u_{ik}, u_{ik})$$

$$= U_i (e_{ik}, u_{ik}, u_{ik})$$

$$= \frac{1}{2} (u_{ik} - u_{ik}^*)^T R_{ii} (u_{ik} - u_{ik}^*) + u_{ik}^T R_{ii} (u_{ik} - u_{ik}^*)$$

(70)

Using (70) into (66) yields

$$J_i^l = Q_i^* (\tilde{e}_{il}, u_{il}^*)$$

$$+ \sum_{k=l}^{\infty} \frac{1}{2} (u_{ik} - u_{ik}^*)^T R_{ii} (u_{ik} - u_{ik}^*)$$

$$+ u_{ik}^T R_{ii} (u_{ik} - u_{ik}^*)$$

(71)

So the optimal performance index $Q_i^* (\tilde{e}_{il}, u_{il}^*)$ (at time index $l$) is given by the unique value such that

$$J_i^l = Q_i^* (\tilde{e}_{il}, u_{il}^*)$$

(72)

c) From (70), we can observe

$$\sum_{k=1}^{\infty} U_i (e_{ik}, u_{ik}, u_{ik}) - U_i^* (e_{ik}, u_{ik}, u_{ik}) > 0.$$ 

Then

$$J_i^l \leq J_i$$

According to Definition 3, Nash equilibrium exists in multi-agent systems.
REFERENCES

[1] K. G. Vamvoudakis, F. L. Lewis, and G. R. Hadas, “Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality,” Automatica, vol. 48, no. 8, pp. 1598–1611, Aug. 2012.

[2] D. Wang, D. Liu, Q. Wei, D. Zhao, and N. Jin, “Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming,” Automatica, vol. 48, no. 8, pp. 1825–1832, Aug. 2012.

[3] Z. Zhang, D. Zhao, J. Gao, D. Wang, and Y. Dai, “FMRQ—A multiagent reinforcement learning algorithm for fully cooperative tasks,” IEEE Trans. Cybern., vol. 47, no. 6, pp. 1367–1379, Jun. 2017.

[4] R. Song, Q. Wei, H. Zhang, and F. L. Lewis, “Discrete-time Non-Zero-Sum games with completely unknown dynamics,” IEEE Trans. Cybern., early access, Dec. 31, 2020, doi: 10.1109/TCYB.2019.2957406.

[5] R. Song, J. Li, and F. L. Lewis, “Robust optimal control for disturbed nonlinear zero-sum differential games based on single NN and least squares,” IEEE Trans. Syst., Man, Cybern. Syst., vol. 50, no. 11, pp. 4009–4019, Nov. 2020.

[6] M. I. Abouheaf, F. L. Lewis, K. G. Vamvoudakis, S. Haesaert, and R. Babuska, “Multi-agent discrete-time graphical games and reinforcement learning solutions,” Automatica, vol. 50, no. 12, pp. 3038–3053, Dec. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109814004282

[7] K. G. Vamvoudakis and F. L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878–888, May 2010.

[8] M. Dastani and L. van der Torre, “Programming BOID-Plan Agents deliberating about conflicts among defeasible mental attitudes and plans,” in Proc. 3rd Int. Joint Conf. Auton. Agents Multiagent Syst., Jul. 2004, pp. 706–713.

[9] K.-K. Oh, M.-C. Park, and H.-S. Ahn, “A survey of multi-agent formation control,” Automatica, vol. 53, pp. 424–440, Mar. 2015.

[10] Q. Li and Z.-P. Jiang, “Global analysis of multi-agent systems based on Viscsek’s model,” IEEE Trans. Autom. Control, vol. 54, no. 12, pp. 2876–2881, Dec. 2009.

[11] R. S. Sutton and A. G. Barto, Reinforcement Learning, 2nd ed. Cambridge, MA, USA: MIT Press, 2019.

[12] X. Gan, H. Guo, and Z. Li, “A new multi-agent reinforcement learning method based on evolving dynamic correlation matrix,” IEEE Access, vol. 7, pp. 162127–162138, 2019.

[13] R. Babuska, “Multi-agent discrete-time graphical games and reinforcement learning solutions,” Automatica, vol. 50, no. 12, pp. 3038–3053, Dec. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109814004282

[14] K. G. Vamvoudakis and F. L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878–888, May 2010.

[15] M. Dastani and L. van der Torre, “Programming BOID-Plan Agents deliberating about conflicts among defeasible mental attitudes and plans,” in Proc. 3rd Int. Joint Conf. Auton. Agents Multiagent Syst., Jul. 2004, pp. 706–713.

[16] K.-K. Oh, M.-C. Park, and H.-S. Ahn, “A survey of multi-agent formation control,” Automatica, vol. 53, pp. 424–440, Mar. 2015.

[17] Q. Li and Z.-P. Jiang, “Global analysis of multi-agent systems based on Viscsek’s model,” IEEE Trans. Autom. Control, vol. 54, no. 12, pp. 2876–2881, Dec. 2009.

[18] R. S. Sutton and A. G. Barto, Reinforcement Learning, 2nd ed. Cambridge, MA, USA: MIT Press, 2019.

[19] X. Gan, H. Guo, and Z. Li, “A new multi-agent reinforcement learning method based on evolving dynamic correlation matrix,” IEEE Access, vol. 7, pp. 162127–162138, 2019.

[20] A. E. Bryson, “Optimal control-1950 to 1985,” Automatica, vol. 50, no. 12, pp. 3038–3053, Dec. 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109814004282

[21] K. G. Vamvoudakis and F. L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878–888, May 2010.

[22] M. Dastani and L. van der Torre, “Programming BOID-Plan Agents deliberating about conflicts among defeasible mental attitudes and plans,” in Proc. 3rd Int. Joint Conf. Auton. Agents Multiagent Syst., Jul. 2004, pp. 706–713.

[23] K. G. Vamvoudakis and F. L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878–888, May 2010.

[24] A. E. Bryson, “Optimal control-1950 to 1985,” IEEE Control Syst., vol. 16, no. 3, pp. 26–33, Jun. 1996.

[25] H. Zhang, H. Jiang, Y. Luo, and G. Xiao, “Data-driven optimal consensus control for discrete-time multi-agent systems with unknown dynamics using reinforcement learning method,” IEEE Trans. Ind. Electron., vol. 64, no. 5, pp. 4091–4100, May 2017.

[26] W. Qiu, Z. Zhang, and F. L. Lewis, “Multi-agent reinforcement learning for fully cooperative multi-agent graphical games,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2018, pp. 1–6.

[27] Z. Peng, J. Hu, K. Shi, R. Luo, R. Huang, B. K. Ghosh, and J. Huang, “A novel optimal bipartite consensus control scheme for unknown multi-agent systems via model-free reinforcement learning,” Appl. Math. Comput., vol. 369, Mar. 2020, Art. no. 124821. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000634952030136

[28] Y. Ben, X. F. Huang, Z. Y. Zhuang, R. Yan, and S. Xu, “Agent-based approach for crowded pedestrian evacuation simulation,” IET Intell. Transp. Syst., vol. 7, no. 1, pp. 56–67, 2013.