Advances in Reinforcement Learning Structures for Continuous-time Dynamical Systems

Bahare Kiumarsi, *Student Member, IEEE,* Kyriakos G. Vamvoudakis, *Senior Member, IEEE,* Hamidreza Modares, *Member, IEEE,* and Frank L. Lewis, *Fellow, IEEE*

***Abstract*—This paper presents some new adaptive control structures based on reinforcement learning for computing online the solutions to optimal tracking control problems and multi-player differential games. We design a new family of adaptive controllers that converge in real time to optimal control and game theoretic solutions by using data measured along the system trajectories. This is a new approach to data-driven optimization. Integral reinforcement learning is used to develop policy iteration based algorithms that find optimal solutions online and do not require full knowledge of the system dynamics. A new experience replay technique is given that uses past data for present learning and significantly speeds up convergence. A new method of off-policy learning allows learning of optimal solutions without knowing any dynamic information. New algorithm will be presented for solving online the non zero-sum multi-player games for continuous-time systems. Each player maintains two adaptive learning structures, a critic network and an actor network. The result is an adaptive control system that learns based on the interplay of agents in a game, to deliver true online gaming behavior.**

I. Introduction

Optimal feedback control design has been responsible for much of the successful performance of engineered systems in aerospace, manufacturing, industrial processes, vehicles, ships, robotics, and elsewhere since the 1960s. However, it is difficult to perform optimal designs for nonlinear process systems since they rely on off-line solutions to complicated Hamilton-Jacobi-Bellman (HJB) equations. Moreover, optimal design generally requires that the full system dynamics be known, which is seldom the case in practical applications [1].

B. Kiumarsi is with UTA Research Institute, University of Texas at Arlington, Fort Worth, Texas 76118, USA, email: kiumarsi@uta.edu.

K. G. Vamvoudakis, is with the Kevin T. Crofton Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA 24061-0203 USA, e-mail: kyriakos@vt.edu.

H. Modares is with the Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, Missouri, 65401 USA, e-mail:modaresh@mst.edu.

F. L. Lewis is with UTA Research Institute, University of Texas at Arlington, Fort Worth, Texas, USA and Qian Ren Consulting Professor, State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang, China, email: lewis@uta.edu.

Reinforcement learning (RL) and adative dynamic programming [2]–[7] methods have been successfully applied to find the solution to the HJB online without requiring complete knowledge about the system dynamics for both continuous-time (CT) and discrete-time (DT) systems [8]–[14]. Efficient off-policy RL algorithms were first presented in [15], [16] that do not need any knowledge about the system dynamics to find the solution to the HJB equation. Most existing RL algorithms are desigend for solving optimal regulation problems. In most real world applications, it is desired the outputs or states of the system track a desired reference trajectory. In [17]–[19], RL algorithms are presented to find the online solution to the optimal tracking problem for CT systems. The online actor-critic approximators have been introduced and further developed by Werbos [20] to implement the presented RL algorithms for nonlinear systems. The critic approximator estimates the value function and is updated to minimize the Bellman equation error. The actor approximator represents a control policy and is updated to minimize the value function.

A persistence of excitation (PE) condition is required to be satisfied to guaranttee convergence to a near optimal solution. However, traditional PE conditions are often difficult or impossible to check online. Also, due to the requirement for the PE condition, the existing RL-based algorithms for CT systems are sample inefficient, that is, they require many samples in order to learn the optimal policy. In order to reduce sample complexity and use available data more effectively, the experience replay technique has been proposed in the context of RL for [21], [22]. In this technique, a number of recent samples are stored in a database and they are presented repeatedly to the underlying RL algorithm.

In this paper, we present online adaptive controllers based on policy iteration (PI) algorithms, namely the on-policy and off-policy integral RL (IRL), that converge to the optimal solution of HJB equation arising in optimal tracking control problems without requiring any knowledge about the system dynamics or reference trajectory. Moreover, the online solution of the multiplayer non

zero-sum games using on-policy IRL algorithm is considered. The algorithms are implemented using online actor-critic structure and experience replay technique.

II. Optimal Tracking Control Problem

Consider the nonlinear time-invariant system given by,

$$\dot{x}(t) = f(x(t)) + g(x(t))u(t), \tag{1}$$

where $x \in \mathbb{R}^n$ is a measurable state vector, $u(t) \in \mathbb{R}^m$ is the control input, $f(x) \in \mathbb{R}^n$ is the drift dynamics and $g(x) \in \mathbb{R}^{n\times m}$ is the input dynamics. It is assumed that $f(0) = 0$ and $f(x) + g(x)u$ is locally Lipschitz and the system is stabilizable.

The goal of the optimal tracking control problem is to design the optimal control policy $u^\star(t)$ so as to make the system (1) track a desired trajectory $x_d(t)$ in an optimal manner by minimizing a predefined performance function.

The desired trajectory is generated by command generator model

$$\dot{x}_d(t) = h_d(x_d(t)), \tag{2}$$

where $x_d(t) \in \mathbb{R}^n$.

Define the tracking error as

$$e(t) = x(t) - x_d(t).$$

A. Standard solution to the optimal tracking problem

In the tracking problem, the control input consists of two terms: a feedforward term which guarantees perfect tracking and a feedback term which stabilizes the system dynamics. Most solutions to the nonlinear optimal tracking problems find the feedforward term using the dynamics inversion concept and the feedback term by solving an HJB equation.

The feedforward term can be obtained using dynamics inversion concept as

$$u_d(t) = g(x_d)^{-1}\left(\dot{x_d} - f(x_d)\right).$$

The feedback term is designed in an optimal manner by minimizing the following performance function

$$J(e(t), u_e(t)) = \int_t^\infty \left(e(\tau)^T Q_e\, e(\tau) + {u_e}^T R_e\, u_e\right)\, d\tau,$$

where $Q_e \geq 0$, $R_e = R_e^T > 0$. The feedback term of the control input is found by solving the stationarity condition $\partial J(e, u_e)/\partial u_e = 0$ as

$$u_e^\star(t) = -\frac{1}{2} R_e^{-1} g^T \frac{\partial J(e(t))}{\partial e(t)}.$$

Then, the optimal control input including both feedback and feedforward terms is

$$u^\star(t) = u_d(t) + u_e^\star(t).$$

The RL algorithms can be used to find the feedback part of the control input $u_e^\star(t)$ without requiring complete knowledge of the system dynamics. However, obtaining the feedforward part of the control input $u_d(t)$ needs complete knowledge of the system dynamics and the reference trajectory dynamics, which cancels the usefulness of the RL techniques.

B. New formulation to find the solution of the optimal tracking problem

In this section, a new formulation is developed that gives both feedback and feedforward parts of the control input simultaneously and thus enables RL algorithms to solve the tracking problems without requiring complete knowledge of the system dynamics [17], [18].

The augmented system can be constructed in terms of the tracking error $e(t)$ and the reference trajectory $x_d(t)$ as

$$\dot{X}(t) = \begin{bmatrix} \dot{e}(t) \\ \dot{x}_d(t) \end{bmatrix} = \begin{bmatrix} f(e(t) + x_d(t)) - h_d(x_d(t)) \\ h_d(x_d(t)) \end{bmatrix} + \begin{bmatrix} g(e(t) + x_d(t)) \\ 0 \end{bmatrix} u(t) \equiv F(X(t)) + G(X(t))u(t), \tag{3}$$

where the augmented state is

$$X(t) = \begin{bmatrix} e(t) \\ x_d(t) \end{bmatrix}.$$

The performance function is defined as

$$J(x(t), x_d(t), u(t)) = \int_t^\infty e^{-\eta(\tau - t)} \times \left((x(\tau) - x_d(\tau))^T Q\,(x(\tau) - x_d(\tau)) + u^T(\tau) R\, u(\tau)\right) d\tau,$$

where $Q \geq 0$ and $R = R^T > 0$. The value function in terms of the states of the augmented system is written as

$$V(X(t)) = \int_t^\infty e^{-\eta(\tau - t)} r(X, u)\, d\tau = \int_t^\infty e^{-\eta(\tau - t)} \left(X^T(\tau) Q_T\, X(\tau) + u^T(\tau) R\, u(\tau)\right) d\tau, \tag{4}$$

where $Q_T = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}$ and $\eta \geqslant 0$ is the discount factor.

This formulation converts the optimal tracking problem to solving the optimal regulation problem by minimizing the value function (4) subject to the augmented system (3) which gives both feedback and feedforward parts of the control input simultaneously.

A differential equivalent to this is the Bellman equation,

$$r(X,u) - \eta\, V + \nabla V_X^T\big(F(X) + G(X)\, u\big) = 0, V(0) = 0, \tag{5}$$

and the Hamiltonian is given by,

$$H(X,u,\nabla V_X) = r(X,u) - \eta\, V + \nabla V_X^T\big(F(X) + G(X)\, u\big).$$

The optimal value is given by the Bellman optimality equation

$$r(X,u^\star) - \eta\, V^\star + \nabla V_X^{\star T}\big(F(X) + G(X)u^\star\big) = 0, \tag{6}$$

which is just the CT HJB tracking equation. The optimal control is then given as

$$u^\star(t) = \arg\min_u \big(r(X,u^\star) - \eta\, V^\star + \nabla V_X^{\star T}\big(F(X) + G(X)u^\star\big)\big)$$
$$= -\frac{1}{2}R^{-1}G^T(X)\nabla V_X^\star. \tag{7}$$

For the linear system case, considering a quadratic value function, the equivalent of the HJB equation is the well known algebraic Riccati equation (ARE). Solving the HJB equation is generally difficult as it is a nonlinear differential equation, quadratic in the cost function, which also requires complete knowledge of the system dynamics. The following on-policy and off-policy IRL algorithms are presented to approximate the solution of the HJB tracking equation by iterating on the Bellman equation.

III. On-policy Integral Reinforcement Learning

An equivalent formulation of the Bellman equation (5) that does not involve the dynamics is found to be,

$$V(X(t)) = \int_t^{t+T} e^{-\eta\,(\tau-t)}\big(X^T(\tau)\, Q_T(X(\tau)) + u^T(\tau)\, R\, u(\tau)\big)d\tau + e^{-\eta T}V(X(t+T)).$$

for any time $t \geqslant 0$ and time interval $T > 0$. This equation is called the IRL Bellman equation [8], [23]. The following offline PI algorithm can be implemented by iterating on the above IRL Bellman equation and updating the control policy.

Algorithm 1: on-policy IRL algorithm to find the solution of HJB

1: **procedure**
2: Given admissible policy u_0
3: for $j = 0, 1, \ldots$ given u_j and solve for the value $V_{j+1}(X)$ using Bellman equation

$$V_{j+1}(X(t)) = \int_t^{t+T} e^{-\eta\,(\tau-t)}\big(X^T(\tau)Q_T\,X(\tau) + u_j^T(\tau)Ru_j(\tau)\big)d\tau + e^{-\eta T}V_{j+1}(X(t+T)),$$

on convergence, set $V_{j+1}(X) = V_j(X)$
4: Update the control policy $u_{j+1}(t)$ using

$$u_{j+1}(t) = -\frac{1}{2}R^{-1}G^T(X)(\nabla V_X)_{j+1}.$$

5: go to 3
6: **end procedure**

A. Synchronous actor-critic structure to implement on-policy IRL algorithm

In this section, an online solution to the HJB tracking equation is presented which only requires partial knowledge about the system dynamics. The learning structure uses the value function approximation with two neural networks (NNs), namely an actor and a critic. Instead of sequentially updating the critic and actor NNs, as in Algorithm 1, both are updated simultaneously in real time. We call this synchronous online PI.

Assume that the value function is a smooth function, there exists a single-layer NN such that the $V(X)$ can be uniformly approximated as

$$V(X) = W_1^T\phi_1(X) + \epsilon(X), \forall X.$$

where $\phi_1(X) : \mathbb{R}^n \to \mathbb{R}^N$ is the basis function vector, N is the number of basis functions, and $\epsilon(X)$ is the approximation error.

The wights of critic NN, W_1 , are unknown. Then, it is approximated in real time as

$$\hat{V}(X) = \hat{W}_c^T\phi_1(X),\ \forall X,$$

where $\hat{W}_c$ are the current estimated values of the ideal critic approximator weights W_1. Therefore, the IRL tracking Bellman equation becomes

$$e_B = \hat{W}_c^T\Delta\phi_1(t) + \int_{t-T}^{t} e^{-\eta(\tau-t+T)}\big(X^T\,Q_T X + \hat{u}^T R\hat{u}\big)d\tau,$$

where $\Delta\phi_1(t) = e^{-\eta T}\phi_1(t) - \phi_1(t-T)$ and $e_B \in \mathbb{R}^n$ is temporal difference (TD) error after using current critic approximator weights and $\hat{u}$ is given by,

$$\hat{u}(X) = -\frac{1}{2}R^{-1}G^T(X)\frac{\partial\phi_1}{\partial X}^T\hat{W}_u,$$

where $\hat{W}_u$ is the current estimated value of the optimal actor weight.

The critic NN weights are updated to minimize $E_B = \frac{1}{2}e_B^2$.

The tuning laws for the critic weights are selected as the gradient descent algorithm, that is

$$\dot{\hat{W}}_c = -\alpha_c \frac{\Delta\phi_1(t)}{\left(\Delta\phi_1(t)^T \Delta\phi_1(t) + 1\right)^2} e_B,$$

where $\alpha_c > 0$ is the learning rate. The weights for the actor $\hat{W}_u$ need to be picked in order to guarantee closed-loop stability. Hence one has,

$$\dot{\hat{W}}_u = -\alpha_u \big((F_2 \hat{W}_u - F_1 \Delta\phi_1(t)^T \hat{W}_c) - \frac{1}{4} (\frac{\partial \phi_1}{\partial X} G(X) \times R^{-1} G(X)^T \frac{\partial \phi_1}{\partial X}^T) \hat{W}_u \big(\frac{\Delta\phi_1(t)^T}{\left(\Delta\phi_1(t)^T \Delta\phi_1(t)) + 1\right)^2} \big) \hat{W}_c \big),$$

where $\alpha_u > 0$ is a tuning gain and $F_1, F_2 > 0$ are user defined positive definite matrices picked appropriately for stability.

B. Tuning the weights of the critic NN using experience replay learning technique

To speed up and obtain an easy-to-check condition for the convergence of the IRL algorithm, the recent transition samples are stored and repeatedly presented to the gradient-based update rule. This is a gradient-descent algorithm that not only minimizes the instantaneous TD error, but also minimizes the TD errors for the stored transition samples [24], [25]. The samples are stored in a history stack. To collect data in the history stack, consider $\Delta\phi_1(t_j)$ as evaluated values of $\Delta\phi_1$ at the recorded time t_j. Then, define the Bellman equation error (TD error) at the recorded time t_j using the current critic weights estimation $\hat{W}_c$ as

$$\begin{aligned}(e_B)_j =& \hat{W}_c^T \Delta\phi_1(t_j) \\ &+ \int_{t_j - T}^{t_j} e^{\eta(\tau - t_j + T)} \left(X^T Q_T X + \hat{u}^T R \hat{u} \right) d\tau \end{aligned} \quad (8)$$

The proposed novel experience replay based gradient-decent algorithm for the critic NN is now given as

$$\begin{aligned}\dot{\hat{W}}_c =& -\alpha_c \frac{\Delta\phi_1(t)}{\left(\Delta\phi_1(t)^T \Delta\phi_1(t) + 1\right)^2} e_B \\ &- \alpha_c \sum_{j=1}^{l} \frac{\Delta\phi_1(t_j)}{\left(\Delta\phi_1(t_j)^T \Delta\phi_1(t_j) + 1\right)^2} (e_B)_j \end{aligned}$$

IV. Off-policy Integral Reinforcement Learning

In order to develop the off-policy IRL algorithm, the system dynamics (3) is rewritten as [16], [26]

$$\dot{X}(t) = F(X(t)) + G(X(t))\, u_j(t) + G(X(t))\, (u(t) - u_j(t)), \quad (9)$$

where $u_j(t)$ is the policy to be updated. By contrast, $u(t)$ is the behavior policy that are actually applied to the system dynamics to generate the data. Differentiating $V_j(X)$ along with the system dynamics (9) and using $u_{j+1}(t) = -\frac{1}{2} R^{-1} G^T(X) (\nabla V_X)_j$ gives

$$\begin{aligned}\dot{V}_j =& (\nabla V_X{}^T)_j (F + G\, u_j) + (\nabla V_X{}^T)_j G (u - u_i) = \eta\, V_j \\ &- X^T Q_T\, X - u_j{}^T R\, u_j - 2\, u_{j+1}{}^T R\, (u - u_j). \end{aligned} \quad (10)$$

Multiplying both sides of (10) by $e^{-\eta(\tau - t)}$ and integrating from both sides yields the following off-policy IRL Bellman equation

$$\begin{aligned}& e^{-\eta T} V_j(X(t+T)) - V_j(X(t)) = \\ & \int_t^{t+T} e^{-\eta(\tau - t)} (-X^T Q_T\, X - u_j{}^T R\, u_j)\, d\tau + \\ & \int_t^{t+T} e^{-\eta(\tau - t)} (-2\, u_{j+1}{}^T R\, (u - u_j))\, d\tau. \end{aligned} \quad (11)$$

Note that for a fixed control policy u (the policy which is applied to the system), (11) can be solved for both value function V_j and updated policy u_{j+1}, simultaneously. It is shown in [26] that the off-policy IRL equation (11) gives the same solution for the value function as the HJB equation (6) and the same updated control policy as (7).

The following algorithm uses the off-policy tracking Bellman equation (11) to iteratively solve the HJB equation (6) without requiring any knowledge of the system dynamics.

Algorithm 2: off-policy IRL algorithm to find the solution of HJB

1: **procedure**
2: Given admissible policy $u(t)$
3: for $j = 0, 1, \ldots$ given u_j and solve for the value V_j and u_{j+1} using off-policy Bellman equation

$$\begin{aligned}& e^{-\eta T} V_j(X(t+T)) - V_j(X(t)) = \\ & \int_t^{t+T} e^{-\eta(\tau - t)} \left(-Q(X) - u_j^T R u_j - 2\, u_{j+1}{}^T R\, (u - u_j) \right) d\tau, \end{aligned}$$

on convergence, set $V_{j+1} = V_j$
4: go to 3
5: **end procedure**

To implement off-policy IRL Algorithm 2, the actor-critic structure is used to approximate the value function and control policy [16].

V. Non Zero-sum Games

This section presents the formulation of N-player games for nonlinear systems. RL is used to find the Nash equilibrium of non-cooperative games online in real time using measured data.

Consider the N-player nonlinear time-invariant differential game,

$$\dot{x}(t) = f(x(t)) + \sum_{j=1}^{N} g_j(x(t))u_j(t),$$

where $x \in \mathbb{R}^n$ is a measurable state vector, $u_j(t) \in \mathbb{R}^{m_j}$ are the control inputs, $f(x) \in \mathbb{R}^n$, is the drift dynamics, $g_j(x) \in \mathbb{R}^{n\times m_j}$ is the input dynamics. It is assumed that $f(0) = 0$ and $f(x) + \sum_{j=1}^{N} g_j(x)u_j$ is locally Lipschitz and that the system is stabilizable.

The performance function associated with each player is given by,

$$J_i(x(0), u_1, u_2, \cdots, u_N) = \int_0^\infty \big(r_i(x, u_1, \cdots, u_N)\big)dt$$
$$\equiv \int_0^\infty \big(Q_i(x) + \sum_{j=1}^{N} u_j^T R_{ij} u_j\big)dt, \ \forall i \in \mathcal{N},$$

where $Q_i(\cdot) \geq 0$ is generally nonlinear and $R_{ii} > 0, \forall i \in \mathcal{N}$, $R_{ij} \geq 0, \ \forall j \neq i \in \mathcal{N}$ are symmetric matrices and $\mathcal{N} := \{1, 2, \cdots, N\}$.

The value can be defined as,

$$V_i(x, u_1, u_2, \cdots, u_N) =$$
$$\int_t^\infty \big(r_i(x, u_1, \cdots, u_N)\big)d\tau, \ \forall i \in \mathcal{N}, \forall x, u_1, u_2, \cdots, u_N.$$

The Bellman equation for each player is given as,

$$r_i(x, u_1, \cdots, u_N) + \nabla(V_x^T)_i\big(f(x) + \sum_{j=1}^{N} g_j(x)u_j\big) = 0,$$
$$V_i(0) = 0, \forall i \in \mathcal{N}.$$

The Hamiltonian functions is defined as,

$$H_i(x, u_1, \cdots, u_N, \nabla(V_x^T)_i) = r_i(x, u_1, \cdots, u_N) + \nabla(V_x^T)_i$$
$$\big(f(x) + \sum_{j=1}^{N} g_j(x)u_j\big), \forall i \in \mathcal{N}.$$

The associated feedback control policies are given by,

$$u_i^\star = \arg\min_{u_i} H_i(x, u_1, \cdots, u_N, \nabla(V_x^{\star T})_i)$$
$$= -\frac{1}{2}R_{ii}^{-1} g_i^T(x)\nabla(V_x)_i, \forall i \in \mathcal{N}.$$

After substituting the feedback control policies into the Hamiltonian one has the coupled HJ equations,

$$0 = Q_i(x) + \frac{1}{4}\sum_{j=1}^{N} \nabla(V_x^T)_j g_j(x) R_{jj}^{-T} R_{ij} R_{jj}^{-1} g_j^T(x)\nabla(V_x)_j$$
$$+ \nabla(V_x^T)_i\big(f(x) - \frac{1}{2}\sum_{j=1}^{N} g_j(x)R_{jj}^{-1} g_j^T(x)\nabla(V_x)_j\,\partial x\big), \forall i \in \mathcal{N}. \quad (12)$$

To approximate the solution to the coupled HJ equations, a PI algorithm is now developed as follows by iterating on the Bellman equation.

Algorithm 3: PI for Regulation in Non-Zero Sum Games

1: **procedure**
2: Given N-tuple of admissible policies $\mu_i^k(0), \ \forall i \in \mathcal{N}$
3: **while** $\|V_i^{\mu^{(k)}} - V_i^{\mu^{(k-1)}}\| \geq \epsilon_{\text{iac}}, \ \forall i \in \mathcal{N}$ **do**
4: Solve for the N-tuple of costs $V_i^k(x)$ using the coupled Bellman equations

$$Q_i(x) + \frac{\partial V_i^k}{\partial x}^T \big(f(x) + \sum_{i=1}^{N} g_i(x)\mu_i^k\big) + \mu_i^{kT} R_{ii}\mu_i^k$$
$$+ \sum_{j=1}^{N} \mu_j^{kT} R_{ij}\mu_j^k = 0, V^{\mu_i^k}(0) = 0.$$

5: Update the N-tuple of control policies $\mu_i^{k+1}, \ \forall i \in \mathcal{N}$ using

$$\mu_i^{k+1} = -\frac{1}{2}R_{ii}^{-1} g_i^T(x)\frac{\partial V_i^k}{\partial x}^T.$$

6: $k = k + 1$
7: **end while**
8: **end procedure**

The PI Algorithm 3 requires complete knowledge of the system dynamics. To obviate this requirement, IRL can be used as follows. An equivalent formulation of the coupled IRL Bellman equation that does not involve the dynamics is given as,

$$V_i(x(t-T)) = \int_{t-T}^{t} r_i(x(\tau), u_1(\tau), \cdots, u_N(\tau))d\tau$$
$$+ V_i(x(t)), \ \forall i \in \mathcal{N},$$

for any time $t \geqslant 0$ and time interval $T > 0$.

Let the value function for each player be approximated as

$$\hat{V}_i(x) = \hat{W}_{ic}^T \phi_i(x), \ \forall x, \ i \in \mathcal{N}.$$

By using a similar procedure as Section III, the update laws can be rewritten as,

$$\dot{W}_{ic} = -\alpha_i \frac{\Delta\phi_i(t)}{\big(\Delta\phi_i(t))^T\Delta\phi_i(t) + 1\big)^2}\big(\Delta\phi_i(t)^T\hat{W}_{ic} +$$
$$\int_{t-T}^{t} \big(Q_i(x) + \hat{u}_i^T R_{ii}\hat{u}_i + \sum_{j=1}^{N} \hat{u}_j^T R_{ij}\hat{u}_j\big)d\tau\big), \forall i \in \mathcal{N},$$

and,

$$\dot{\hat{W}}_{iu} = -\alpha_{iu}\big((F_i\hat{W}_{iu} -$$
$$L_i\Delta\phi_i(t)^T\hat{W}_{ic}) - \frac{1}{4}\sum_{j=1}^{N}(\frac{\partial\phi_i}{\partial x}g_i(x)R_{ii}^{-T}R_{ij}R_{ii}^{-1}g_i(x)^T\frac{\partial\phi_i}{\partial x}^T)$$
$$\hat{W}_{iu}\frac{\Delta\phi_i(t)^T}{\big(\Delta\phi_i(t)^T\Delta\phi_i(t) + 1\big)^2}\hat{W}_{jc}\big), \ \forall i \in \mathcal{N},$$

respectively with $\Delta\phi_i(t) := \phi_i(t) - \phi_i(t-T)$.

A model-free learning algorithm for non-zero sum Nash games has been investigated in [27]. System identification is used in [28] to avoid knowing the complete knowledge of the system dynamics.

VI. Conclusion

In this paper, we have presented the online solution to optimal tracking control of nonlinear continuous-time systems and multi-player differential games using RL algorithms. The proposed algorithms use the measured data of the system to find the optimal solution without requiring any knowledge about the system dynamics or reference trajectory. The future work is to extend presented results to control of distributed multi-agent systems with nonlinear dynamics. Moreover, deep learning can be integrated with the presented results to deal with large scale systems with a huge amount of data.

References

[1] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*. John Wiley, 2012.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1. MIT press Cambridge, 1998.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic Programming*. Athena Scientific, MA, 1996.

[4] W. B. Powell, *Approximate Dynamic Programming*. Hoboken, NJ: Wiley, 2007.

[5] H. Zhang, D. Liu, Y. Luo, and D. Wang, *Adaptive Dynamic Programming for Control*. Springer-Verlag London, 2013.

[6] K. Vamvoudakis, P. Antsaklis, W. Dixon, J. Hespanha, F. Lewis, H. Modares, and B. Kiumarsi, "Autonomy and machine intelligence in complex systems: A tutorial," in *American Control Conference (ACC), 2015*, pp. 5062–5079, July 2015.

[7] D. Liu, Q. Wei, D. Wang, X. Yang, and H. Li, *Adaptive Dynamic Programming with Application in Optimal Control*. Springer International Publishing, 2017.

[8] D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Networks*, vol. 22, no. 3, pp. 237 – 246, 2009.

[9] S. Bhasin, R. Kamalapurkar, M. Johnson, K. G. Vamvoudakis, F. L. Lewis, and W. E. Dixon, "A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, no. 1, pp. 89–92, 2013.

[10] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 4, pp. 943–949, 2008.

[11] T. Dierks and S. Jagannathan, "Online optimal control of nonlinear discrete-time systems using approximate dynamic programming," *Journal of Control Theory and Applications*, vol. 9, no. 3, pp. 361–369, 2011.

[12] P. He and S. Jagannathan, "Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 425–436, 2007.

[13] H. Li, D. Liu, and D. Wang, "Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 706–714, 2014.

[14] Y. Luo and H. Zhang, "Approximate optimal control for a class of nonlinear discrete-time systems with saturating actuators," *Progress in Natural Science*, vol. 18, no. 8, pp. 1023 – 1029, 2008.

[15] Y. Jiang and Z.-P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, 2012.

[16] Y. Jiang and Z. P. Jiang, "Robust adaptive dynamic programming and feedback stabilization of nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 882–893, 2014.

[17] H. Modares and F. L. Lewis, "Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 3051–3056, 2014.

[18] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, no. 7, pp. 1780–1792, 2014.

[19] K. G. Vamvoudakis, H. Modares, B. Kiumarsi, and F. L. Lewis, "Game theory-based control system algorithms with real-time reinforcement learning: How to solve multiplayer games online," *IEEE Control Systems*, vol. 37, no. 1, pp. 33–52, 2017.

[20] P. J. Werbos, *A Menu of Designs for Reinforcement Learning Over Time*. MIT press, 1991.

[21] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193 – 202, 2014.

[22] R. Kamalapurkar, J. Klotz, and W. E. Dixon, "Concurrent learning-based online approximate feedback nash equilibrium solution of N-player nonzero-sum differential games," *IEEE/CAA journal of Automatica Sinica*, vol. 1, no. 3, p. 239247, 2014.

[23] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477 – 484, 2009.

[24] R. Kamalapurkar, P. Walters, and W. E. Dixon, "Model-based reinforcement learning for approximate optimal regulation," *Automatica*, vol. 64, p. 94104, 2016.

[25] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193 – 202, 2014.

[26] H. Modares, F. L. Lewis, and Z. P. Jiang, "H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2550–2562, 2015.

[27] K. G. Vamvoudakis, "Non-zero sum nash Q-learning for unknown deterministic continuous-time linear systems," *Automatica*, vol. 61, pp. 274–281, 2015.

[28] M. Johnson, R. Kamalapurkar, S. Bhasin, and W. E. Dixon, "Approximate N-player nonzero-sum game solution for an uncertain continuous nonlinear system," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 8, pp. 1645–1658, 2015.