CHAPTER

ADAPTIVE H_∞ TRACKING CONTROL OF NONLINEAR SYSTEMS USING REINFORCEMENT LEARNING

14

Hamidreza Modares[*], **Bahare Kiumarsi**[†], **Kyriakos G. Vamvoudakis**[‡], **Frank L. Lewis**[†,§]

[*]*Missouri University of Science and Technology, Rolla, MO, United States* [†]*UTA Research Institute, University of Texas at Arlington, Fort Worth, TX, United States* [‡]*Virginia Tech, Blacksburg, VA, United States* [§]*State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang, China*

CONTENTS

CHAPTER POINTS

- The result of this approach is to design an online data-based solution to the H_∞ tracking control problem.
- Reinforcement learning is employed to learn the solution to the H_∞ tracking in real-time and without requiring the system dynamics.

Adaptive Learning Methods for Nonlinear System Modeling. DOI: 10.1016/B978-0-12-812976-0.00018-X

14.1 INTRODUCTION

Reinforcement learning (RL) [1–3], inspired by learning mechanisms observed in animals, is concerned with how an agent or decision maker takes actions so as to optimize a cost of its long-term interactions with the environment. The cost function is prescribed and captures some desired system behaviors such as minimizing the transient error and minimizing the control effort for achieving a specific goal. The agent learns an optimal policy so that, by taking actions produced based on this policy, the long-term cost function is optimized. Similar to RL, optimal control involves finding an optimal policy by optimizing a long-term performance criterion. Strong connections between RL and optimal control have prompted a major effort towards introducing and developing online and model-free RL algorithms to learn the solution to optimal control problems [4–6].

RL methods have been successfully used to solve the optimal regulation problems by learning the solution to the so-called Hamilton–Jacobi equations arising from both optimal H_2 [7–18] and H_∞ [19–30] regulation problems. For continuous-time (CT) systems, [8,9] proposed a promising RL algorithm, called integral RL (IRL), to learn the solution to the Hamilton–Jacobi–Bellman (HJB) equations using only partial knowledge about the system dynamics. They used an iterative online policy iteration [31] procedure to implement their IRL algorithm. The original IRL algorithm and many of its extensions are on-policy algorithms. That is, the policy that is applied to the system to generate data for learning (behavior policy) is the same as the policy that is being updated and learned about (target policy). The work [15] presented an off-policy RL algorithms for CT systems in which the behavior policy could be different from the target policy. This algorithm does not require any knowledge of the system dynamics and is data efficient because it reuses the data generated by the behavior policy to learn as many target policies as required. Many variants and extensions of off-policy RL algorithms are presented in the literature. Other than the IRL-based PI algorithms and off-policy RL algorithms, efficient synchronous PI algorithms with guaranteed closed-loop stability were proposed for CT systems in [7,11,12] to learn the solution to the HJB equation. Synchronous IRL algorithms were also presented for solving the HJB equation in [23,32].

Although RL algorithms have been widely used to solve the optimal regulation problems, few results considered solving the optimal tracking control problem (OTCP) for both discrete-time [33–36] and continuous-time systems [6,37]. Moreover, existing methods for continuous-time systems require the exact knowledge of the system dynamics *a priori* while finding the feedforward part of the control input using either the dynamic inversion concept or the solution of output regulator equations [39–41]. While the importance of the RL algorithms is well understood for solving optimal regulation problems for uncertain systems, the requirement of the exact knowledge of the system dynamics for finding the steady-state part of the control input in the existing OTCP formulation does not allow for direct extending of the IRL algorithm for solving the OTCP.

In this chapter, we develop adaptive optimal controllers based on the RL techniques to learn the optimal H_∞ tracking control solutions for nonlinear continuous-time systems without knowing the system dynamics or the command generator dynamics. An augmented system is first constructed from the tracking error dynamics and the command generator dynamics to introduce a new discounted performance function for the OTCP. The tracking Hamilton–Jacobi–Isaac (HJI) equations are then derived to solve OTCPs. Off-policy RL algorithms, implemented on an actor-critic structure, are used to find the solution to the tracking HJI equations online using only measured data along the augmented system trajectories. These algorithms are developed for both affine and nonaffine nonlinear systems. Therefore, they can be employed in control of many real-world applications, including robot manipulators, mobile robots, unmanned aerial vehicles (UAVs), power systems and human–robot interaction systems.

14.2 H_∞ OPTIMAL TRACKING CONTROL FOR NONLINEAR AFFINE SYSTEMS

Existing solutions to the H_∞ tracking problem are composed of two steps [38–41]. A feedforward control input is designed to guarantee perfect tracking using either dynamic inversion or by solving the so-called output regulator equations in the first step. A feedback control input is designed in the second step by solving an HJI equation to stabilize the tracking error dynamics. In these methods, procedures for computing the feedback and feedforward terms are based on offline solution methods which require complete knowledge of the system dynamics. In this section, a new formulation for the H_∞ tracking is presented which allows developing model-free RL solutions.

Consider the nonlinear time-invariant system given as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\,\mathbf{u}(t) + \mathbf{k}(\mathbf{x}(t))\,\mathbf{w}(t), \tag{14.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$, $\mathbf{u}(t) \in \mathbb{R}^m$ and $\mathbf{w}(t) \in \mathbb{R}^p$ represent the state of the system, the control input and the external disturbance of the system, respectively. The drift dynamics is represented by $\mathbf{f}(\mathbf{x}(t)) \in \mathbb{R}^n$, $\mathbf{g}(\mathbf{x}(t)) \in \mathbb{R}^{n\times m}$ is the input dynamics and $\mathbf{k}(\mathbf{x}(t)) \in \mathbb{R}^p$ is the disturbance dynamics. It is assumed that $\mathbf{f}(\mathbf{0}) = \mathbf{0}$ and $\mathbf{f}(\mathbf{x}(t))$, $\mathbf{g}(\mathbf{x}(t)$ and $\mathbf{k}(\mathbf{x}(t))$ are unknown Lipschitz functions and the system is stabilizable.

Assumption 1. Let $\mathbf{r}(t)$ be the bounded reference trajectory and assume that there exists a Lipschitz continuous command generator function $\mathbf{h_d}(t) \in \mathbb{R}^n$ with $\mathbf{h_d}(\mathbf{0}) = \mathbf{0}$ such that

$$\dot{\mathbf{r}}(t) = \mathbf{h_d}(t)\,\mathbf{r}(t). \tag{14.2}$$

Define the tracking error

$$\mathbf{e_d}(t) \triangleq \mathbf{x}(t) - \mathbf{r}(t). \tag{14.3}$$

Using (14.1)–(14.3), the tracking error dynamics is given by

$$\dot{\mathbf{e}}_\mathbf{d}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\,\mathbf{u}(t) + \mathbf{k}(\mathbf{x}(t))\,\mathbf{w}(t) - \mathbf{h_d}(\mathbf{r}(t)). \tag{14.4}$$

The performance output to be controlled is defined such that it satisfies

$$\|z(t)\|^2 = \mathbf{e_d}^T\,\mathbf{Q}\,\mathbf{e_d} + \mathbf{u}^T\,\mathbf{R}\,\mathbf{u}. \tag{14.5}$$

The goal of the H_∞ tracking is to attenuate the effect of the disturbance input $\mathbf{w}$ on the performance output $\mathbf{z}$. Before defining the H_∞ tracking control problem, we define the following general L_2-gain or disturbance attenuation condition.

Definition 1 (Bounded L_2-gain or disturbance attenuation)**.** The nonlinear system (14.1) is said to have L_2-gain less than or equal to γ if the following disturbance attenuation condition is satisfied for all $\mathbf{w} \in L_2[0, \infty)$:

$$\frac{\int_t^\infty e^{-\alpha(\tau-t)} \|\mathbf{z}(\tau)\|^2 \, d\tau}{\int_t^\infty e^{-\alpha(\tau-t)} \|\mathbf{w}(\tau)\|^2 d\tau} \leqslant \gamma^2, \tag{14.6}$$

where $\alpha > 0$ is the discount factor and γ represents the amount of attenuation from the disturbance input $\mathbf{w}(t)$ to the defined performance output variable $\mathbf{z}(t)$.

The disturbance attenuation condition (14.6) implies that the effect of the disturbance input to the desired performance output is attenuated by a degree at least equal to γ. The desired performance output represents a meaningful cost in the sense that it includes a positive penalty on the tracking error and a positive penalty on the control effort. The use of the discount factor is essential. This is because the feedforward part of the control input does not converge to zero in general and thus penalizing the control input in the performance function without a discount factor makes the performance function unbounded.

Using (14.5) in (14.6) one has

$$\int_t^\infty e^{-\alpha(\tau-t)} (\mathbf{e_d}^T \mathbf{Q} \mathbf{e_d} + \mathbf{u}^T \mathbf{R} \mathbf{u}) d\tau \leqslant \gamma^2 \int_t^\infty e^{-\alpha(\tau-t)} (\mathbf{w}^T \mathbf{w}) \, d\tau. \tag{14.7}$$

Definition 2 (H_∞ optimal tracking)**.** The H_∞ tracking control problem is to find a control policy $\mathbf{u} = \beta(\mathbf{e_d}, \mathbf{r})$ for some smooth function β depending on the tracking error $\mathbf{e}$ and the reference trajectory $\mathbf{r}$, such that:
(i) The closed-loop system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x}) \, \beta(\mathbf{e_d}, \mathbf{r}) \; + \mathbf{k}(\mathbf{x}) \, \mathbf{w}$ satisfies the attenuation condition (14.7).
(ii) The tracking error dynamics (14.4) with $\mathbf{w} = 0$ is locally asymptotically stable.

The main difference between Definition 2 and the standard definition of the H_∞ tracking control problem (see [38], Definition 5.2.1) is that a more general disturbance attenuation condition is defined here. Previous work on the H_∞ optimal tracking divides the control input into feedback and feedforward parts. The feedforward part is first obtained separately without considering any optimality criterion. Then, the problem of optimal design of the feedback part is reduced to an H_∞ optimal regulation problem. In contrast, in the new formulation, both feedback and feedforward parts of the control input are obtained simultaneously and optimally as a result of the defined L_2-gain with discount factor in (14.7).

14.2.1 HJI EQUATION FOR H_∞ OPTIMAL TRACKING

In this section, it is first shown that the problem of solving the H_∞ tracking problem can be transformed into a min–max optimization problem subject to an augmented system composed of the tracking error dynamics and the command generator dynamics. A tracking HJI equation is then developed which gives the solution to the min–max optimization problem. The stability and L_2-gain boundedness of the tracking HJI control solution are discussed.

Define the augmented system state

$$\mathbf{X}(t) = [\mathbf{e_d}(t)^T \; \mathbf{r}(t)^T]^T \in \mathbb{R}^{2n},$$

where $\mathbf{e_d}(t)$ is the tracking error defined in (14.3) and $\mathbf{r}(t)$ is the reference trajectory.

Using (14.2) and (14.4), define the augmented system

$$\dot{\mathbf{X}}(t) = \mathbf{F}(\mathbf{X}(t)) + \mathbf{G}(\mathbf{X}(t))\,\mathbf{u}(t) + \mathbf{K}(\mathbf{X}(t))\,\mathbf{w}(t), \tag{14.8}$$

where $\mathbf{u}(t) = \mathbf{u}(\mathbf{X}(t))$ and

$$\mathbf{F}(\mathbf{X}) = \begin{bmatrix} \mathbf{f}(\mathbf{e_d}+\mathbf{r}) - \mathbf{h_d}(\mathbf{r}) \\ \mathbf{h_d}(\mathbf{r}) \end{bmatrix}, \mathbf{G}(\mathbf{X}) = \begin{bmatrix} \mathbf{g}(\mathbf{e_d}+\mathbf{r}) \\ \mathbf{0} \end{bmatrix}, \mathbf{K}(\mathbf{X}) = \begin{bmatrix} \mathbf{k}(\mathbf{e_d}+\mathbf{r}) \\ \mathbf{0} \end{bmatrix}.$$

The disturbance attenuation condition (14.7) using the augmented state becomes

$$\int_t^\infty e^{-\alpha(\tau-t)}(\mathbf{X}^T\mathbf{Q_T}\mathbf{X} + \mathbf{u}^T\mathbf{R}\mathbf{u})d\tau \leqslant \gamma^2 \int_t^\infty e^{-\alpha(\tau-t)}(\mathbf{w}^T\mathbf{w})\,d\tau, \tag{14.9}$$

where

$$\mathbf{Q_T} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

Based on (14.9), define the performance function

$$J(\mathbf{u},\mathbf{w}) = \int_t^\infty e^{-\alpha(\tau-t)}(\mathbf{X}^T\mathbf{Q_T}\,\mathbf{X} + \mathbf{u}^T\mathbf{R}\,\mathbf{u} - \gamma^2\mathbf{w}^T\mathbf{w})\,d\tau. \tag{14.10}$$

Solvability of the H_∞ control problem is equivalent to solvability of the following zero-sum game [42]:

$$V^\star(\mathbf{X}(t)) = J(\mathbf{u}^\star,\mathbf{w}^\star) = \min_{\mathbf{u}}\max_{\mathbf{d}}\, J(\mathbf{u},\mathbf{w}), \tag{14.11}$$

where J is defined in (14.10) and $V^\star(\mathbf{X}(t))$ is defined as the optimal value function. This two-player zero-sum game control problem has a unique solution if a game theoretic saddle point exists, *i.e.*, if the following Nash condition holds:

$$V^\star(\mathbf{X}(t)) = \min_{\mathbf{u}}\max_{\mathbf{d}}\, J(\mathbf{u},\mathbf{w}) = \max_{\mathbf{d}}\min_{\mathbf{u}}\, J(\mathbf{u},\mathbf{w}).$$

Differentiating (14.10), note that $V(\mathbf{X}(t)) = J(\mathbf{u}(t),\mathbf{w}(t))$ gives the following Bellman equation:

$$H(V,\mathbf{u},\mathbf{w}) \triangleq \mathbf{X}^T\mathbf{Q_T}\mathbf{X} + \mathbf{u}^T\mathbf{R}\,\mathbf{u} - \gamma^2\mathbf{w}^T\mathbf{w} - \alpha V + {V_\mathbf{X}}^T(\mathbf{F} + \mathbf{G}\,\mathbf{u} + \mathbf{K}\,\mathbf{w}) = 0, \tag{14.12}$$

where $\mathbf{F} \triangleq \mathbf{F}(\mathbf{X})$, $\mathbf{G} \triangleq \mathbf{G}(\mathbf{X})$, $\mathbf{K} \triangleq \mathbf{K}(\mathbf{X})$ and $V_\mathbf{X} = \partial V/\partial \mathbf{X}$.

Applying stationarity conditions $\partial H(V^\star,\mathbf{u},\mathbf{w})/\partial\mathbf{u} = 0$, $\partial H(V^\star,\mathbf{u},\mathbf{w})/\partial\mathbf{w} = 0$ [43] gives the optimal control and disturbance inputs as

$$\mathbf{u}^\star = -\frac{1}{2}\mathbf{R}^{-1}\mathbf{G}^T\,{V_\mathbf{X}}^\star, \tag{14.13}$$

$$\mathbf{w}^\star = \frac{1}{2\gamma^2}\mathbf{K}^T\,{V_\mathbf{X}}^\star, \tag{14.14}$$

where $V^\star$ is the optimal value function defined in (14.11). Substituting the control input (14.13) and the disturbance (14.14) into (14.12), the following tracking HJI equation is obtained:

$$\begin{aligned} H(V^\star, \mathbf{u}^\star, \mathbf{w}^\star) &\triangleq \mathbf{X}^T \mathbf{Q_T} \mathbf{X} + V_{\mathbf{X}}^{\star T} \mathbf{F} - \alpha V_{\mathbf{X}} \\ &- \frac{1}{4} V_{\mathbf{X}}^{\star T} \mathbf{G}^T \mathbf{R}^{-1} \mathbf{G} V_{\mathbf{X}}^\star + \frac{1}{4\gamma^2} V_{\mathbf{X}}^{\star T} \mathbf{K} \mathbf{K^T} V_{\mathbf{X}}^\star = 0. \end{aligned} \tag{14.15}$$

It is shown in [44] that the control solution (14.13)–(14.15) satisfies the disturbance attenuation condition (14.9) (part (i) of Definition 2) and that it guarantees the stability of the tracking error dynamics (14.4) without the disturbance (part (ii) of Definition 2), if the discount factor is less than an upper bound.

14.2.2 OFF-POLICY IRL FOR LEARNING THE TRACKING HJI EQUATION

In this section, an off-policy RL algorithm is first given to learn this control solution online and without requiring any knowledge of the system dynamics.

The Bellman equation (14.12) is linear in the cost function V, while the HJI equation (14.15) is nonlinear in the value function $V^\star$. Therefore, solving the Bellman equation for V is easier than solving the HJI for $V^\star$. Instead of directly solving for $V^\star$, a policy iteration (PI) algorithm iterates on both control and disturbance players to break the HJI equation into a sequence of differential equations linear in the cost. An offline PI algorithm for solving the H_∞ optimal tracking problem is given as follows.

Algorithm 1 Offline RL algorithm.

```
1: procedure
2:     Start with an admissible stabilizing control policy $\mathbf{u}^0$.
3:     For a control input $\mathbf{u}^j$ and disturbance policy $\mathbf{w}^j$, find $V^j$ using the following Bellman equation:

       $H(V^j,\ \mathbf{u}^j, \mathbf{w}^j) = \mathbf{X}^T \mathbf{Q_T} \mathbf{X} + (V_{\mathbf{X}}^j)^T (\mathbf{F} + \mathbf{G}\mathbf{u}^j + \mathbf{K}\mathbf{w}^j) - \alpha V^j + (\mathbf{u}^j)^T \mathbf{R}\mathbf{u}^j - \gamma^2 (\mathbf{w}^j)^T \mathbf{w}^j = 0.$   (14.16)

4:     Update the disturbance using

       $\mathbf{w}^{j+1} = \arg\max_{\mathbf{d}} \left[H(V^j, \mathbf{u}^j, \mathbf{w})\right] = \frac{1}{2\gamma^2} \mathbf{K}^T V_{\mathbf{X}}^j$   (14.17)

   and the control policy using

       $\mathbf{u}^{j+1} = \arg\min_{\mathbf{u}} \left[H(V^j, \mathbf{u}, \mathbf{w}^{j+1})\right] = -\frac{1}{2} \mathbf{R}^{-1} \mathbf{G}^T V_{\mathbf{X}}^j.$   (14.18)

5:     Go to 3.
6: end procedure
```

Algorithm 1 extends the results of the simultaneous RL algorithm in [27] to the tracking problem. The convergence of this algorithm to the minimal nonnegative solution of the HJI equation was shown in [27]. In fact, similar to [27], the convergence of Algorithm 1 can be established by proving that iteration on (14.16) is essentially a Newton iterative sequence which converges to the unique solution of the HJI equation (14.15).

Algorithm 1 requires complete knowledge of the system dynamics. In the following, the off-policy IRL algorithm, which was presented in [14,15] for solving the H_2 optimal regulation problem, is extended here to solve the H_∞ optimal tracking for systems with completely unknown dynamics. To this end, the system dynamics (14.8) is first written as

$$\dot{\mathbf{X}} = \mathbf{F} + \mathbf{G}\,\mathbf{u}^j + \mathbf{K}\,\mathbf{w}^j + \mathbf{G}\,(\mathbf{u} - \mathbf{u}^j) + \mathbf{K}\,(\mathbf{w} - \mathbf{w}^j), \tag{14.19}$$

where $\mathbf{u}^j \in \mathbb{R}^m$ and $\mathbf{w}^j \in \mathbb{R}^q$ are policies to be updated. In this equation, the control input $\mathbf{u}$ is the behavior policy which is applied to the system to generate data for learning, while $\mathbf{u}^j$ is the target policy which is evaluated and updated using data generated by the behavior policy. The fixed control policy $\mathbf{u}$ should be a stable and exploring control policy. Moreover, the disturbance input $\mathbf{w}$ is the actual external disturbance that comes from an external source and is not under our control. However, the disturbance $\mathbf{w}^j$ is the disturbance that is evaluated and updated. One advantage of this off-policy IRL Bellman equation is that, in contrast to on-policy RL-based methods, the disturbance input that is applied to the system does not require to be adjustable.

Differentiating $V^j(\mathbf{X})$ along with the system dynamics (14.19) and using (14.16)–(14.18) gives

$$\begin{aligned}\dot{V}^j &= (V_{\mathbf{X}}{}^j)^T(\mathbf{F} + \mathbf{G}\,\mathbf{u}^j + \mathbf{K}\,\mathbf{w}^j) + (V_{\mathbf{X}}{}^j)^T\mathbf{G}(\mathbf{u} - \mathbf{u}^j) + (V_{\mathbf{X}}{}^j)^T\mathbf{K}\,(\mathbf{w} - \mathbf{w}^j)\\ &= \alpha\, V^j - \mathbf{X}^T\mathbf{Q_T}\,\mathbf{X} - (\mathbf{u}^j)^T\mathbf{R}\,\mathbf{u}^j + \gamma^2(\mathbf{w}^j)^T\mathbf{w}^j - \\ &\quad 2\,(\mathbf{u}^{j+1})^T\mathbf{R}\,(\mathbf{u} - \mathbf{u}^j) + 2\gamma^2(\mathbf{w}^{j+1})^T(\mathbf{w} - \mathbf{w}^j).\end{aligned} \tag{14.20}$$

Multiplying both sides of (14.20) by $e^{-\alpha(\tau - t)}$ and integrating from both sides yields the following off-policy IRL Bellman equation:

$$\begin{aligned}&e^{-\alpha T}V^j(\mathbf{X}(t+T)) - V^j(\mathbf{X}(t)) =\\ &\int_t^{t+T} e^{-\alpha(\tau - t)}(-\mathbf{X}^T\mathbf{Q_T}\,\mathbf{X} - (\mathbf{u}^j)^T\mathbf{R}\,\mathbf{u}^j + \gamma^2(\mathbf{w}^j)^T\mathbf{w}^j)\,d\tau\\ &+ \int_t^{t+T} e^{-\alpha(\tau - t)}(-2\,(\mathbf{u}^{j+1})^T\mathbf{R}\,(\mathbf{u} - \mathbf{u}^j) + 2\gamma^2(\mathbf{w}^{j+1})^T(\mathbf{w} - \mathbf{w}^j))\,d\tau.\end{aligned} \tag{14.21}$$

Note that, for a fixed control policy $\mathbf{u}$ (the policy that is applied to the system) and a given disturbance $\mathbf{w}$ (the actual disturbance that is applied to the system), Eq. (14.21) can be solved for both the value function V^j and the updated policies $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$ simultaneously.

Lemma 1. *The off-policy IRL equation* (14.21) *gives the same solution for the value function as the Bellman equation* (14.16) *and the same updated control and disturbance policies as* (14.18) *and* (14.17).

Proof. See [44]. □

The following algorithm uses the off-policy tracking Bellman equation (14.21) to iteratively solve the HJI equation (14.15) without requiring any knowledge of the system dynamics. The implementation of this algorithm is discussed in the next subsection. It is shown how the data collected from a fixed control policy u are reused to evaluate many updated control policies $\mathbf{u}_i$ sequentially until convergence to the optimal solution is achieved.

Algorithm 2 Online off-policy RL algorithm for solving the tracking HJI equation.

```
1: procedure
2:     Phase 1 (data collection using a fixed control policy): Apply a fixed control policy $\mathbf{u}$ to the system and collect required system information about the state, control input and disturbance at $N$ different sampling intervals $T$.
3:     For a control input $\mathbf{u}^j$ and disturbance policy $\mathbf{w}^j$, find $V^j$ using the following Bellman equation:
           $H(V^j, \mathbf{u}^j, \mathbf{w}^j) = \mathbf{X}^T \mathbf{Q_T} \mathbf{X} + (V_{\mathbf{X}}^{j})^T (\mathbf{F} + \mathbf{G}\mathbf{u}^j + \mathbf{K}\mathbf{w}^j) - \alpha V^j + (\mathbf{u}^j)^T \mathbf{R}\mathbf{u}^j - \gamma^2 (\mathbf{w}^j)^T \mathbf{w}^j = 0.$   (14.22)
4:     Phase 2 (reuse of collected data sequentially to find an optimal policy iteratively): Given $\mathbf{u}^j$ and $\mathbf{w}^j$, use collected information in phase 1 to Solve the following Bellman equation for $V^j$, $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$ simultaneously:
           $e^{-\alpha T} V^j(\mathbf{X}(t+T)) - V^j(\mathbf{X}(t)) =$
           $\int_t^{t+T} e^{-\alpha(\tau - t)} (-\mathbf{X}^T \mathbf{Q_T} \mathbf{X} - (\mathbf{u}^j)^T \mathbf{R}\mathbf{u}^j + \gamma^2 (\mathbf{w}^j)^T \mathbf{w}^j)\, d\tau$   (14.23)
           $+ \int_t^{t+T} e^{-\alpha(\tau - t)} (-2(\mathbf{u}^{j+1})^T \mathbf{R}(\mathbf{u} - \mathbf{u}^j) + 2\gamma^2 (\mathbf{w}^{j+1})^T (\mathbf{w} - \mathbf{w}^j))\, d\tau.$
5:     Stop if a stopping criterion is met, otherwise set $j = j + 1$ and go to 3.
6: end procedure
```

Inspired by the off-policy algorithm in [14], Algorithm 2 has two separate phases. First, a fixed initial exploratory control policy $\mathbf{u}$ is applied and the system information is recorded over the time interval T. Second, without requiring any knowledge of the system dynamics, the information collected in phase 1 is repeatedly used to find a sequence of updated policies $\mathbf{u}^j$ and $\mathbf{w}^j$ converging to $\mathbf{u}^\star$ and $\mathbf{w}^\star$. Note that Eq. (14.23) is a scalar equation and can be solved in a least square sense after collecting enough data samples from the system. It is shown in the following section how to collect required information in phase 1 and reuse it in phase 2 in a least square sense to solve (14.23) for V^j, $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$ simultaneously. After the learning is done and the optimal control policy $\mathbf{u}^\star$ is found, it can be applied to the system.

Theorem 1 (Convergence of Algorithm 2)**.** *The off-policy Algorithm 2 converges to the optimal control and disturbance solutions given by* (14.13) *and* (14.14) *where the value function satisfies the tracking HJI equation* (14.15).

Proof. See [44]. □

14.2.3 IMPLEMENTING ALGORITHM 2 USING NEURAL NETWORKS

In order to implement the off-policy RL Algorithm 2, it is required to reuse the collected information found by applying a fixed control policy $\mathbf{u}$ to the system to solve Eq. (14.23) for V^j, $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$

iteratively. Three neural networks (NNs), *i.e.*, the actor NN, the critic NN and the disturber NN, are used here to approximate the value function and the updated control and disturbance policies in the Bellman equation (14.23). That is, the solution V^j, $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$ of the Bellman equation (14.23) is approximated by three NNs as

$$\hat{V}^j(\mathbf{X}) = \hat{\mathbf{W}}_\mathbf{1}^T \sigma(\mathbf{X}), \tag{14.24}$$

$$\hat{\mathbf{u}}^{j+1}(\mathbf{X}) = \hat{\mathbf{W}}_\mathbf{2}^T \phi(\mathbf{X}), \tag{14.25}$$

$$\hat{\mathbf{w}}^{j+1}(\mathbf{X}) = \hat{\mathbf{W}}_\mathbf{3}^T \varphi(\mathbf{X}), \tag{14.26}$$

where $\sigma = [\sigma_1, ..., \sigma_{l_1}] \in \mathbb{R}^{l_1}$, $\phi = [\phi_1, ..., \phi_{l_2}] \in \mathbb{R}^{l_2}$ and $\varphi = [\varphi_1, ..., \varphi_{l_3}] \in \mathbb{R}^{l_3}$ provide suitable basis function vectors, $\hat{\mathbf{W}}_\mathbf{1} \in \mathbb{R}^{l_1}$, $\hat{\mathbf{W}}_\mathbf{2} \in \mathbb{R}^{m \times l_2}$ and $\hat{\mathbf{W}}_\mathbf{3} \in \mathbb{R}^{q \times l_3}$ are constant weight vectors and l_1, l_2 and l_3 are the number of neurons. Define $\mathbf{v}^1 = [v_1^1, ..., v_1^m]^T = \mathbf{u} - \mathbf{u}^j$, $\mathbf{v}^2 = [v_1^2, ..., v_q^2]^T = \mathbf{w} - \mathbf{w}^j$ and assume $\mathbf{R} = diag(r, ..., r_m)$. Then, substituting (14.24)–(14.26) in (14.23) yields

$$\begin{aligned} e(t) = {} & \hat{\mathbf{W}}_1^T (e^{-\alpha T} \sigma(\mathbf{X}(t+T)) - \sigma(\mathbf{X}(t))) \\ & - \int_t^{t+T} e^{-\alpha(\tau - t)} (-\mathbf{X}^T \mathbf{Q_T} \mathbf{X} - (\mathbf{u}^j)^T \mathbf{R} \mathbf{u}^j + \gamma^2 (\mathbf{w}^j)^T \mathbf{w}^j) d\tau \\ & + 2 \sum_{l=1}^m r_l \int_t^{t+T} e^{-\alpha(\tau - t)} \hat{\mathbf{W}}_{2,l}^T \phi(\mathbf{X}(t)) v_l^1 d\tau \\ & - 2\gamma^2 \sum_{k=1}^q \int_t^{t+T} e^{-\alpha(\tau - t)} \hat{\mathbf{W}}_{3,k}^T \varphi(\mathbf{X}(t)) v_k^2 d\tau, \end{aligned} \tag{14.27}$$

where $e(t)$ is the Bellman approximation error, $\hat{\mathbf{W}}_\mathbf{2,l}$ is the lth column of $\hat{\mathbf{W}}_\mathbf{2}$ and $\hat{\mathbf{W}}_\mathbf{3,k}$ is the kth column of $\hat{\mathbf{W}}_\mathbf{3}$. The Bellman approximation error is the continuous-time counterpart of the temporal difference (TD) [10]. In order to bring the TD error to its minimum value, the least squares method is used. To this end, rewrite Eq. (14.27) as

$$y(t) + e(t) = \hat{\mathbf{W}}^\mathbf{T} \mathbf{h}(\mathbf{t}), \tag{14.28}$$

where

$$\hat{\mathbf{W}} = [\hat{\mathbf{W}}_\mathbf{1}^T, \hat{\mathbf{W}}_\mathbf{2,1}^T, ..., \hat{\mathbf{W}}_\mathbf{2,m}^T, \hat{\mathbf{W}}_\mathbf{3,1}^T, ..., \hat{\mathbf{W}}_\mathbf{3,q}^T]^T \in \mathbb{R}^{l_1 + m \times l_2 + q \times l_3},$$

$$\mathbf{h}(\mathbf{t}) = \begin{bmatrix} e^{-\alpha T} \sigma(\mathbf{X}(t+T)) - \sigma(\mathbf{X}(t))) \\ 2r_1 \int_t^{t+T} e^{-\alpha(\tau - t)} \phi(\mathbf{X}(t)) v_1^1 d\tau \\ \vdots \\ 2r_m \int_t^{t+T} e^{-\alpha(\tau - t)} \phi(\mathbf{X}(t)) v_m^1 d\tau \\ -2\gamma^2 \int_t^{t+T} e^{-\alpha(\tau - t)} \varphi(\mathbf{X}(t)) v_1^2 d\tau \\ \vdots \\ -2\gamma^2 \int_t^{t+T} e^{-\alpha(\tau - t)} \varphi(\mathbf{X}(t)) v_q^2 d\tau \end{bmatrix}, \tag{14.29}$$

$$y(t) = \int_t^{t+T} e^{-\alpha(\tau - t)}(-\mathbf{X}^T \mathbf{Q_T} \mathbf{X} - (\mathbf{u}^j)^T \mathbf{R} \mathbf{u}^j + \gamma^2 (\mathbf{w}^j)^T \mathbf{w}^j) d\tau. \tag{14.30}$$

The parameter vector $\hat{\mathbf{W}}$, which gives the approximated value function, actor and disturbance (14.24)–(14.26), is found by minimizing, in the least squares sense, the Bellman error. Assume that the systems state, input and disturbance information are collected at $N \geqslant l_1 + m \times l_2 + q \times l_3$ (the number of independent elements in $\hat{\mathbf{W}}$) points t_1 to t_N in the state space, over the same time interval T in phase 1. Then, for a given $\mathbf{u}^j$ and $\mathbf{w}^j$, one can use this information to evaluate (14.29) and (14.30) at N points to form

$$\mathbf{H} = [\mathbf{h}(t_1), ..., \mathbf{h}(t_N)],$$
$$\mathbf{Y} = [y(t_1), ..., y(t_N)]^T.$$

The least squares solution to (14.28) is then equal to

$$\hat{\mathbf{W}} = (\mathbf{H}\mathbf{H}^T)^{-1}\mathbf{H}\mathbf{Y},$$

which gives V^j, $\mathbf{u}^{j+1}$ and $\mathbf{w}^{j+1}$. Note that although $\mathbf{X}(t+T)$ appears in Eq. (14.27), this equation is solved in a least square sense after observing N samples $\mathbf{X}(t), \mathbf{X}(t+T), \ldots, \mathbf{X}(t+NT)$. Therefore, the knowledge of the system is not required to predict the future state $\mathbf{X}(t+T)$ at time t to solve (14.27).

14.3 H_∞ OPTIMAL TRACKING CONTROL FOR A CLASS OF NONLINEAR NONAFFINE SYSTEMS

This section considers the design of an RL-based optimal tracking control solution for a class of nonaffine systems.

14.3.1 A CLASS OF NONAFFINE DYNAMICAL SYSTEMS

A special class of nonaffine systems can be described as

$$\dot{\mathbf{X}}(t) = \mathbf{f}(\mathbf{X}(t)) + \mathbf{g}(\mathbf{X}(t))\mathbf{L}(\mathbf{u}) + \mathbf{D}\,\mathbf{w}(t), \tag{14.31}$$

where $\mathbf{X}(t) \in \mathbb{R}^n$, $\mathbf{u}(t) \in \mathbb{R}^m$ and $\mathbf{w}(t) \in \mathbb{R}^p$ are the state of the system, the control input and the external disturbance input, respectively. The functions $\mathbf{f}(\mathbf{X}(t))$ and $\mathbf{g}(\mathbf{X}(t))$ are Lipschitz functions. This system is affine in a nonlinear function $\mathbf{L}(.)$ of the control input $\mathbf{u}(t)$. This class of nonaffine systems allows the definition of a new performance function for the optimal H_∞ problem such that the existence of the constrained optimal control is assured (if any exists).

The following example shows that the UAV as a real-world application can be presented in the form of (14.31).

Example 1. A general class of nonlinear nonaffine UAV systems has the following well-known form:

$$\dot{x}_1 = V \cos\gamma \, \cos\psi + d_1 w_1,$$

$$
\begin{aligned}
\dot{x}_2 &= V\cos\gamma\ \sin\psi + d_2 w_2,\\
\dot{x}_3 &= -V\sin\gamma\ + d_3 w_3,\\
\dot{V} &= -\alpha_2 V^2 - g\sin\gamma + \alpha_1 \bar{T} - \alpha_3 n_z - \alpha_4 \frac{n_z^2}{V^2},\\
\dot{\gamma} &= \frac{g}{V}(n_z\cos\phi - \cos\gamma),\\
\dot{\psi} &= \frac{g}{V\cos\gamma} n_z \sin\phi,
\end{aligned}
\tag{14.32}
$$

with

$$
\begin{aligned}
n_x &= \frac{\bar{T}\bar{T}_{\max}\cos\alpha - D}{mg},\\
n_x &= \frac{\bar{T}\bar{T}_{\max}\sin\alpha + K}{mg},
\end{aligned}
$$

where x_1, x_2, x_3 are the UAV location coordinates, γ is the pitch angle, ψ is the heading angle, ϕ is bank angle, V is the UAV velocity and m is the mass of the UAV. The terms n_x and n_z denote longitudinal and normal components of the load factor, depending on the current thrust $\bar{T}$, drag force D and lift force K (g is the acceleration due to gravity) [45].

Define the state of the UAV as

$$
\mathbf{X} = \{x_1, x_2, x_3, V, \gamma, \psi\}^{\mathrm{T}} \tag{14.33}
$$

and the control input and disturbance inputs (wind velocity) as $\mathbf{u}(t) = [\bar{T}, n_z, \phi]^{\mathrm{T}} = [\begin{matrix} u_1 & u_2 & u_3 \end{matrix}]^{\mathrm{T}}$ and $\mathbf{w}(t)$, respectively. The constraints on the control input are as follows:

$$
\begin{aligned}
|u_1| &\leqslant \bar{u}_1,\\
|u_2| &\leqslant \bar{u}_2.
\end{aligned}
\tag{14.34}
$$

Using (14.32) and (14.33), the UAV dynamics can be written as a nonlinear nonaffine CT system as

$$
\dot{\mathbf{X}}(t) = M(\mathbf{X}(t), \mathbf{u}(t)) + \mathbf{D}\,\mathbf{w}(t), \tag{14.35}
$$

with

$$
\mathbf{D} = [\begin{matrix} d_1 & d_2 & d_3 & 0 & 0 & 0 \end{matrix}]^{\mathrm{T}},
$$

$$
\mathbf{M}(\mathbf{X}, \mathbf{u}) = \begin{bmatrix}
x_4\cos(x_5)\cos(x_6)\\
x_4\cos(x_5)\sin(x_6)\\
-x_4\sin(x_5)\\
-\alpha_2 x_4^2 - g\sin(x_5) + \alpha_1 u_1 - \alpha_3 u_2 - \alpha_4 \frac{u_2^2}{x_4^2}\\
\frac{g}{x_4}(-\cos(x_5) + u_2\cos(u_3))\\
\frac{g}{x_4\cos(x_5)} u_2 \sin(u_3)
\end{bmatrix}.
$$

The UAV dynamics (14.35) can be written in the form of (14.31) with

$$\mathbf{f}(\mathbf{X}(t)) = \begin{bmatrix} x_4\cos(x_5)\cos(x_6) \\ x_4\cos(x_5)\sin(x_6) \\ -x_4\sin(x_5) \\ -\alpha_2 x_4^2 - g\sin(x_5) \\ \frac{g}{x_4}(-\cos(x_5) \\ 0 \end{bmatrix}, \quad \mathbf{g}(\mathbf{X}(t)) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & -\alpha_3 & -\frac{\alpha_4}{x_4^2} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \frac{g}{x_4\cos(x_5)} \end{bmatrix},$$

$$\mathbf{L}(\mathbf{u}(t)) = \begin{bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_2^2 \\ u_2\cos(u_3) \\ u_2\sin(u_3) \end{bmatrix}.$$

Eq. (14.31) represents a large class of nonaffine systems far larger than the systems that are affine in the control itself. In fact, most aircraft dynamics can be expressed in the form of (14.31) if the lift equation satisfies certain assumptions [45].

14.3.2 PERFORMANCE FUNCTION AND H_∞ CONTROL TRACKING FOR NONAFFINE SYSTEMS

It is shown in [46] that the existence of an admissible optimal control solution for nonaffine systems depends on how the utility function $r(\mathbf{X}, \mathbf{u})$ is defined. Moreover, to deal with the input constraints, a nonquadratic performance index needs to be defined as follows.

Let the reference trajectory be generated by the command generator dynamics (14.2). The performance or control output $\mathbf{z}(t)$ is defined such that it satisfies

$$\|\mathbf{z}(t)\|^2 = (\mathbf{X}-\mathbf{r})^{\mathrm{T}}\mathbf{Q}\,(\mathbf{X}-\mathbf{r}) + W(\mathbf{L}(\mathbf{u})), \tag{14.36}$$

where $\mathbf{Q} \succeq 0$ and $W(\mathbf{L}(\mathbf{u}))$ is a positive definite nonquadratic function of $L(\mathbf{u})$ which penalizes the control effort and is chosen as follows to assure the constrained control effort:

$$W(\mathbf{L}(\mathbf{u})) = \int_0^{\mathbf{L}(\mathbf{u})} \mathbf{w}(s)\,ds = \sum_{j=1}^{l}(\int_0^{\mathbf{L}_j(\mathbf{u})} w_j(s_j)\,ds_j),$$

where $\mathbf{w}(s) = \tanh^{-1}(\bar{\mathbf{L}}^{-1}s) = \begin{bmatrix} w_1(s_1) & \cdots & w_l(s_l) \end{bmatrix}^{\mathrm{T}}$ and $\bar{\mathbf{L}}$ is the constant diagonal matrix given by $\bar{\mathbf{L}} = diag(\bar{L}_1, ..., \bar{L}_l)$, which determines the bounds on $\mathbf{L}(\mathbf{u})$. Note that the bounds are originally given for the control input $\mathbf{u}(t)$ itself. However, one can transform these bounds to bounds on $L(\mathbf{u})$.

The H_∞ control is to develop a control input such that (1) the system (1) with $\mathbf{w} = \mathbf{0}$ is asymptotically stable and (2) the L_2 gain condition (14.6) with $\mathbf{z}(t)$ defined in (14.36) is satisfied in the presence of $\mathbf{w} \in L_2[0, \infty)$.

The disturbance attenuation condition is satisfied if the following cost function is nonpositive:

$$J(\mathbf{X}) = \int_t^\infty e^{-\alpha(\tau-t)} \left[(\mathbf{X}-\mathbf{r})^{\mathrm{T}}\mathbf{Q}(\mathbf{X}-\mathbf{r}) \; + W(\mathbf{L}(\mathbf{u})) - \gamma^2 w^{\mathrm{T}} w \right] d\tau. \tag{14.37}$$

14.3.3 SOLUTION OF THE H_∞ CONTROL TRACKING PROBLEM OF NONAFFINE SYSTEMS

Define the tracking error as (14.3). Then, using (14.2) and (14.31), the tracking error dynamics becomes

$$\dot{\mathbf{e}}_{\mathbf{d}}(t) = \dot{\mathbf{X}}(t) - \dot{\mathbf{r}}(t) \; = \mathbf{f}(\mathbf{X}(t)) + \mathbf{g}(\mathbf{X}(t))\mathbf{L}(\mathbf{u}) + \mathbf{D}\,\mathbf{w}(t) - \mathbf{h}_{\mathbf{d}}(\mathbf{r})(t). \tag{14.38}$$

Based on (14.2) and (14.38), an augmented system can be constructed in terms of the tracking error $\mathbf{e}(t)$ and the reference trajectory $\mathbf{r}(t)$ as

$$\begin{aligned}\dot{\mathbf{Z}}(t) &= \begin{bmatrix} \dot{\mathbf{e}}(\mathbf{t}) \\ \dot{\mathbf{r}}(\mathbf{t}) \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{e}(t)+\mathbf{r}(t)) - \mathbf{h}_{\mathbf{d}}(\mathbf{r}(t)) \\ \mathbf{h}_{\mathbf{d}}(\mathbf{r}(t)) \end{bmatrix} + \begin{bmatrix} \mathbf{g}(\mathbf{e}(t)+\mathbf{r}(t)) \\ \mathbf{0} \end{bmatrix} \mathbf{L}(\mathbf{u}) + \begin{bmatrix} \mathbf{D} \\ \mathbf{0} \end{bmatrix} \mathbf{w}(t) \\ &\equiv \mathbf{F}(\mathbf{Z}(t)) + \mathbf{G}(\mathbf{Z}(t))\mathbf{L}(\mathbf{u}) + \mathbf{K}\mathbf{w}(t),\end{aligned} \tag{14.39}$$

where the augmented state is

$$\mathbf{Z}(t) = \begin{bmatrix} \mathbf{e}(t) \\ \mathbf{r}(t) \end{bmatrix}.$$

The performance index (14.37) can be rewritten as

$$J(\mathbf{L}(\mathbf{u}), \mathbf{w}) = \int_t^\infty e^{-\alpha\,(\tau-t)} \left(\mathbf{Z}^T(\tau)\mathbf{Q_1}\,\mathbf{Z}(\tau) + W(\mathbf{L}(\mathbf{u})) - \gamma^2 \mathbf{w}^{\mathrm{T}}\mathbf{w} \right) d\tau, \tag{14.40}$$

with $\mathbf{Q_1} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$.

The H_∞ control problem can be expressed as a two-player zero-sum differential game in which the control effort policy player $\mathbf{L}(\mathbf{u})$ seeks to minimize the value function, while the disturbance policy player $\mathbf{w}(t)$ desires to maximize it. The goal is to find the feedback saddle point $(\mathbf{L}^\star(\mathbf{u}), \mathbf{w}^\star)$ such that [42]

$$V^\star(\mathbf{Z}(t)) = \min_{\mathbf{L}(\mathbf{u})} \max_{\mathbf{w}} J(\mathbf{L}(\mathbf{u}), \mathbf{w}). \tag{14.41}$$

On the basis of (14.40) and noting that $V(\mathbf{Z}(t)) = J(\mathbf{L}(\mathbf{u}), \mathbf{w})$, the H_∞ tracking Bellman equation is

$$\mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}(\mathbf{u})) - \gamma^2\mathbf{w}^{\mathrm{T}}\mathbf{w} - \alpha V(\mathbf{Z}) + \dot{V}(\mathbf{Z}) = 0 \tag{14.42}$$

and the Hamiltonian is given by

$$H(\mathbf{Z}, L(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}}) = \mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}(\mathbf{u})) - \gamma^2\mathbf{w}^{\mathrm{T}}\mathbf{w} - \alpha V(\mathbf{Z}) + V_{\mathbf{Z}}^{\mathrm{T}}(\mathbf{F}(\mathbf{Z}) + \mathbf{G}(\mathbf{Z})\mathbf{L}(\mathbf{u}) + \mathbf{K}\mathbf{w}).$$

Then the optimal control effort $L(\mathbf{u})$ and disturbance input $w(t)$ for the given problem are obtained by employing the stationarity condition

$$\mathbf{L}^\star(\mathbf{u}) = \arg\min_{\mathbf{L}(\mathbf{u})} H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V^*) \triangleq \frac{d\left[\mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}(\mathbf{u})) - \gamma^2\mathbf{w}^{\mathrm{T}}\mathbf{w} - \alpha V^\star + (V_{\mathbf{Z}}^\star)^{\mathrm{T}}\dot{\mathbf{Z}}\right]}{d\,\mathbf{L}(\mathbf{u})},$$

$$\mathbf{w}^\star = \arg\max_{\mathbf{w}} H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V^\star) \triangleq \frac{d\left[\mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}(\mathbf{u})) - \gamma^2\mathbf{w}^{\mathrm{T}}\mathbf{w} - \alpha V^\star + (V_{\mathbf{Z}}^\star)^{\mathrm{T}}\dot{\mathbf{Z}}\right]}{d\,\mathbf{w}},$$

which give

$$\mathbf{L}^\star(\mathbf{u}) = -\bar{\mathbf{L}}\tanh^{\mathrm{T}}(\mathbf{v}^\star), \tag{14.43}$$

$$\mathbf{w}^\star = \frac{1}{2}\gamma^{-2}(V_{\mathbf{Z}}^*)^{\mathrm{T}}\mathbf{K}, \tag{14.44}$$

where

$$\mathbf{v}^\star = (V_{\mathbf{Z}}^\star)^{\mathrm{T}}\mathbf{G}. \tag{14.45}$$

Substituting (14.43) and (14.44) in Bellman equation (14.42) yields the HJI equation

$$\mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}^\star(\mathbf{u})) - \gamma^2(\mathbf{w}^\star)^{\mathrm{T}}\mathbf{w}^\star - \alpha V^\star(\mathbf{Z}) + \dot{V}^\star(Z) = 0. \tag{14.46}$$

To find the optimal control solution, the tracking HJI equation (14.46) could first be solved and then the control effort $\mathbf{L}^\star(\mathbf{u})$ given by (14.43).

Note that the minimization problem (14.41) is defined in terms of $\mathbf{L}(\mathbf{u})$. Under certain conditions, this is equivalent to minimization in terms of $\mathbf{u}(t)$.

Lemma 2. *We have* $\min_{\mathbf{u}} H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}}) = \min_{\mathbf{L}(\mathbf{u})} H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}})$ *if the elements of* $\mathbf{L}(\mathbf{u})$ *are independent.*

Proof. The minimum of $H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}})$ with respect to u is equal to

$$\min_{u}\ H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}}) = (\frac{\partial\mathbf{L}(\mathbf{u})}{\partial\,\mathbf{u}})^T \frac{\partial H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}})}{\partial\mathbf{L}(\mathbf{u})} = 0 \tag{14.47}$$

and the minimum of $H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}})$ with respect to $\mathbf{L}(\mathbf{u})$ is equal to

$$\min_{\mathbf{L}(\mathbf{u})} H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}}) = \frac{d\,H(\mathbf{Z}, \mathbf{L}(\mathbf{u}), \mathbf{w}, V_{\mathbf{Z}})}{d\,\mathbf{L}(\mathbf{u})} = 0. \tag{14.48}$$

Eqs. (14.47) and (14.48) are equivalent if and only if $J = d\mathbf{L}(\mathbf{u})/d\,\mathbf{u}$ is a nonsingular matrix which guarantees the elements of $\mathbf{L}(\mathbf{u})$ are independent [46]. □

Note that if the elements of $\mathbf{L}^\star(\mathbf{u})$ are independent, then the optimal control is given by

$$\mathbf{u}^\star = -\mathbf{L}^{-1}(\bar{L}\tanh^T(\mathbf{v}^\star)), \tag{14.49}$$

thus $\mathbf{L}(\mathbf{u}^\star) = \mathbf{L}^\star(\mathbf{u})$. Otherwise, it is shown in the subsequent sections how to use (14.43) to find $\mathbf{v}^\star$ and $\mathbf{u}^\star$ consequently to assure $\mathbf{L}(\mathbf{u}^\star) = \mathbf{L}^\star(\mathbf{u})$. The next result holds for both independent and dependent $\mathbf{L}(\mathbf{u})$.

Theorem 2 (Solution to bounded L_2 gain problem)**.** *Assume that there exists a continuous-time positive semidefinite solution $V^\star(\mathbf{Z})$ to the tracking HJI equation* (14.46)*. Let $\mathbf{L}^\star(\mathbf{u})$ be given by* (14.43)*. Then $\mathbf{L}^\star(\mathbf{u})$ in* (14.31) *makes the L_2 gain from the disturbance to the performance output less than or equal to γ.*

Proof. See [46]. □

If the elements of $\mathbf{L}(\mathbf{u})$ are independent, then there exists a $\mathbf{u}^\star$ such that $\mathbf{L}(\mathbf{u}^\star) = \mathbf{L}^\star(\mathbf{u})$ and this $\mathbf{u}^\star$ makes the L_2 gain less than or equal to γ. On the other hand, if the elements of $\mathbf{L}^\star(\mathbf{u})$ are dependent, a method of solution is suggested in subsequent sections.

14.3.4 OFF-POLICY REINFORCEMENT LEARNING FOR NONAFFINE SYSTEMS

In this section, the off-policy RL is presented to solve the optimal H_∞ control of nonaffine nonlinear systems. In the proposed method, no knowledge about the system dynamics and the reference trajectory dynamics is needed. Moreover, it does not require an adjustable disturbance input and it avoids bias in finding the value function. Two algorithms are developed for two different cases: (1) for nonaffine systems with independent elements in $\mathbf{L}(\mathbf{u})$ and (2) for nonaffine systems with dependent elements in $\mathbf{L}(\mathbf{u})$. Then the implementation of these two algorithms is given.

The system dynamics (14.39) can be rewritten as

$$\dot{\mathbf{Z}}(t) = \mathbf{F}(\mathbf{Z}(t)) + \mathbf{G}(\mathbf{Z}(t))\mathbf{L}^j(\mathbf{u}) + \mathbf{K}\mathbf{w}^j + \mathbf{G}(\mathbf{Z}(t))(\mathbf{L}(\mathbf{u}) - \mathbf{L}^j(\mathbf{u})) + \mathbf{K}(\mathbf{w} - \mathbf{w}^j), \tag{14.50}$$

where $\mathbf{L}^j(\mathbf{u})$ and $\mathbf{w}^j(t)$ are the policies that are updated. By contrast, $\mathbf{L}(\mathbf{u})$ and $\mathbf{w}(t)$ are the policies that are applied to the system to collect the data.

By the definition, it is easy to see that

$$e^{-\alpha(t_k - t_{k-1})} V^{j+1}(\mathbf{Z}(t_k)) - V^{j+1}(\mathbf{Z}(t_{k-1})) = \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau - t_{k-1})} (V_{\mathbf{Z}}^{j+1})^{\mathrm{T}} \dot{\mathbf{Z}}(t) - \alpha V^{j+1}\, dt. \tag{14.51}$$

Substituting (14.50) into (14.51) yields

$$e^{-\alpha(t_k - t_{k-1})} V^{j+1}(\mathbf{Z}(t_k)) - V^{j+1}(\mathbf{Z}(t_{k-1})) = \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau - t_{k-1})} (V_{\mathbf{Z}}^{j+1})^{\mathrm{T}} \Big[\mathbf{F}(\mathbf{Z}(t)) + \mathbf{G}(\mathbf{Z}(t))\mathbf{L}^j(\mathbf{u})$$
$$+\mathbf{K}\mathbf{w}^j + \mathbf{G}(\mathbf{Z}(t)(\mathbf{L}(\mathbf{u}) - \mathbf{L}^j(\mathbf{u})) + \mathbf{K}(\mathbf{w} - \mathbf{w}^j)\Big]\, dt. \tag{14.52}$$

On the other hand, one has

$$(V_{\mathbf{Z}}^{j+1})^{\mathrm{T}} \Big[\mathbf{F}(\mathbf{Z}) + \mathbf{G}(\mathbf{Z})\mathbf{L}^j(\mathbf{u}) + \mathbf{K}\mathbf{w}^j\Big] = \alpha V^{j+1} - r_a(\mathbf{Z}(t), L^j(\mathbf{u}), \mathbf{w}^j), \tag{14.53}$$

where

$$r_a(\mathbf{Z}(t), \mathbf{L}^j(\mathbf{u}), \mathbf{w}^j) = \mathbf{Z}^{\mathrm{T}}\mathbf{Q_1}\mathbf{Z} + W(\mathbf{L}(\mathbf{u}^j)) - \gamma^2 (\mathbf{w}^j)^{\mathrm{T}} \mathbf{w}^j.$$

Substituting (14.53) into (14.52) yields

$$e^{-\alpha(t_k-t_{k-1})}V^{j+1}(\mathbf{Z}(t_k)) - V^{j+1}(\mathbf{Z}(t_{k-1})) = \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}((V_{\mathbf{Z}}^{j+1})^{\mathrm{T}} \times \Big[\mathbf{G}(\mathbf{Z}(t)(\mathbf{L}(\mathbf{u}) - \mathbf{L}^j(\mathbf{u})) + \mathbf{K}(\mathbf{w}-\mathbf{w}^j)\Big] - r_a(\mathbf{Z}(t), \mathbf{L}^j(\mathbf{u}), \mathbf{w}^j))dt. \tag{14.54}$$

Using (14.43)–(14.45) in (14.54) yields the following off-policy H_∞ Bellman equation:

$$e^{-\alpha(t_k-t_{k-1})}V^{j+1}(\mathbf{Z}(t_k)) - V^{j+1}(\mathbf{Z}(t_{k-1})) = \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}((\mathbf{v}^{j+1}\bar{\mathbf{L}}\big(\tanh^{\mathrm{T}}(\mathbf{v}^j) - \tanh^{\mathrm{T}}(v)\big) + 2\gamma^2\mathbf{w}^{j+1}(\mathbf{w}-\mathbf{w}^j) - r_a(\mathbf{Z}(t), \mathbf{v}^j, \mathbf{w}^j))dt. \tag{14.55}$$

Note that if $\mathbf{v}^j$ and $\mathbf{w}^j$ are given, the unknown functions $V^{j+1}(\mathbf{Z})$, $\mathbf{v}^{j+1}$ and $\mathbf{w}^{j+1}$ can be approximated using (14.55). Then $\mathbf{L}^{j+1}(\mathbf{u})$ is found from $\mathbf{v}^{j+1}$.

The elements of $\mathbf{L}^{j+1}(\mathbf{u})$ can be either dependent or independent. If elements in $\mathbf{L}^{j+1}(\mathbf{u})$ are independent, then the Bellman equation (14.55) can be solved iteratively using stored data to find $\mathbf{L}^\star(\mathbf{u})$ and the optimal control policy is $\mathbf{u}^\star$. The following algorithm shows how to iterate on (14.55) to find the optimal control policy in this case.

Algorithm 3 Online off-policy RL algorithm for nonaffine system with independent elements in $\mathbf{L}(\mathbf{u})$.

```
1: procedure
2:     Start with the control effort $\mathbf{L}(\mathbf{u})$ and disturbance input $\mathbf{w}$ and collect required system information at $N$ different sampling intervals $T$.
3:     Given $\mathbf{v}^j$ and $\mathbf{w}^j$, use collected information in step 2 to solve the following Bellman equation for $V^{j+1}$, $\mathbf{v}^{j+1}$ and $\mathbf{w}^{j+1}$ simultaneously:
           $$e^{-\alpha(t_k-t_{k-1})}V^{j+1}(\mathbf{Z}(t_k)) - V^{j+1}(\mathbf{Z}(t_{k-1})) = \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}((\mathbf{v}^{j+1}\bar{\mathbf{L}}\big(\tanh^{\mathrm{T}}(\mathbf{v}^j) - \tanh^{\mathrm{T}}(\mathbf{v})\big) + 2\gamma^2\mathbf{w}^{j+1}(\mathbf{w}-\mathbf{w}^j) - r_a(\mathbf{Z}(t), \mathbf{v}^j, \mathbf{w}^j))dt. \tag{14.56}$$
4:     Stop if
           $$\left|\mathbf{v}^{j+1} - \mathbf{v}^j\right| \leqslant \varepsilon \quad \text{and} \quad \left|\mathbf{w}^{j+1} - \mathbf{w}^j\right| \leqslant \varepsilon.$$
5:     Otherwise set $j = j + 1$ and go to 3.
6: end procedure
```

Algorithm 3 gives $\mathbf{L}^{j+1}(\mathbf{u})$ and, if the condition of Lemma 2 is satisfied, then the elements of the control input are $\mathbf{u}^{j+1} = -\mathbf{L}^{-1}(\bar{\mathbf{L}}\tanh^{\mathrm{T}}(\mathbf{v}^{j+1}))$. However, if elements in $\mathbf{L}^{j+1}(\mathbf{u})$ are dependent, then the dependency of its elements must be taken into account by encoding equality constraints while solve Eq. (14.55) for $\mathbf{v}^{j+1}$.

To find a form for solution constraints $\mathbf{L}(\mathbf{u})$ if it has dependent elements, consider the UAV system in Example 1 with

$$\mathbf{L}(\mathbf{u}(t)) = \begin{bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_2^2 \\ u_2 \cos(u_3) \\ u_2 \sin(u_3) \end{bmatrix}.$$

Then, the dependency of the elements of $\mathbf{L}(\mathbf{u})$ becomes

$$L_3 = {L_2}^2 = {L_4}^2 + {L_5}^2.$$

This gives the following equality constraints:

$$\bar{L}_3 \tanh(v_3) = (\bar{L}_2 \tanh(v_2))^2 = (\bar{L}_4 \tanh(v_4))^2 + (\bar{L}_5 \tanh(v_5))^2.$$

In general, it is seen that one has a vector of equality functions

$$\mathbf{f}(\mathbf{L}) = [f_1(\mathbf{L}), ..., f_p(\mathbf{L})]^{\mathrm{T}} = \mathbf{0}, \tag{14.57}$$

with p being the number of dependent elements in $\mathbf{L}(\mathbf{u})$. For example for the UAV system, one has $f_1 = \bar{L}_3 \tanh(v_3) - (\bar{L}_2 \tanh(v_2))^2$, $f_2 = (\bar{L}_2 \tanh(v_2))^2 - (\bar{L}_4 \tanh(v_4))^2 - (\bar{L}_5 \tanh(v_5))^2$ and $f_3 = (\bar{L}_3 \tanh(v_3)) - (\bar{L}_4 \tanh(v_4))^2 - (\bar{L}_5 \tanh(v_5))^2$. This constraint must be taken into account when solving (14.55) for $\mathbf{v}$ using NNs.

The following algorithm shows how to find the optimal control solution for the cases where $\mathbf{L}(\mathbf{u})$ has dependent elements. The details of implementation of solving (14.55) for $\mathbf{v}$ while considering the constraint imposed by the independency of elements of $\mathbf{v}$ are presented in the next subsection.

Before proceeding, $\bar{\mathbf{H}}$ is defined as

$$\begin{aligned}\bar{\mathrm{H}} = &\int_{t_{k-1}}^{t_k} e^{-\alpha(\tau - t_{k-1})}\big((\mathbf{v}^{j+1}\bar{\mathbf{L}}(\tanh^{\mathrm{T}}(\mathbf{v}^j) - \tanh^{\mathrm{T}}(\mathbf{v})) + 2\gamma^2 \mathbf{w}^{j+1}(\mathbf{w} - \mathbf{w}^j) - r_a(\mathbf{Z}(t), \mathbf{v}^j, \mathbf{w}^j)\big)dt \\ &- e^{-\alpha(t_k - t_{k-1})} V^{j+1}(\mathbf{Z}(t_k)) + V^{j+1}(\mathbf{Z}(t_{k-1})).\end{aligned}$$

Algorithm 4 Online off-policy RL algorithm for nonaffine system with dependent elements in $\mathbf{L}(\mathbf{u})$.

```
1: procedure
2:     Start with the control effort $\mathbf{L}(\mathbf{u})$ and disturbance input $\mathbf{w}$ and collect required system information at $N$
       different sampling intervals $T$.
3:     Given $\mathbf{v}^j$ and $\mathbf{w}^j$, use collected information in step 2 to solve the following Bellman equation for $V^{j+1}$,
       $\mathbf{v}^{j+1}$ and $\mathbf{w}^{j+1}$ simultaneously:
           $\min \bar{\mathbf{H}}^2 \ s.t. \ \mathbf{f}(\mathbf{v}) = \mathbf{0}.$        (14.58)
4:     Stop if
           $\left|\mathbf{v}^{j+1} - \mathbf{v}^j\right| \leqslant \varepsilon$ and $\left|\mathbf{w}^{j+1} - \mathbf{w}^j\right| \leqslant \varepsilon.$
5:     Otherwise set $j = j + 1$ and go to 3.
6: end procedure
```

The minimum value of $\bar{\mathbf{H}}$ in Algorithm 4 not considering the constraint (14.57) is zero. If this algorithm terminates, so that $\bar{\mathbf{H}} = 0$, then by Theorem 2 the L_2 gain problem is solved and there exists a $\mathbf{u}^\star$ such that $\mathbf{L}(\mathbf{u}^\star) = \mathbf{L}^\star(\mathbf{u})$.

The following subsection shows how to use NNs along with linear and nonlinear LS, respectively, to implement Algorithms 3 and 4.

14.3.5 NEURAL NETWORKS FOR IMPLEMENTATION OF OFF-POLICY RL ALGORITHMS

In this subsection, the solution of the off-policy H_∞ Bellman equations (14.56) and Eq. (14.58) in Algorithms 3 and 4 using three NNs is presented. The unknown functions $V^{j+1}(Z)$, $\mathbf{v}^{j+1}$ and $\mathbf{w}^{j+1}$ can be approximated by three NNs as

$$\hat{V}^{j+1}(\mathbf{Z}) = \sum_{i=1}^{N_1} \hat{c}_i^{j+1} \phi_i(\mathbf{Z}) = \hat{\mathbf{C}}^{j+1} \phi(\mathbf{Z}), \tag{14.59}$$

$$\hat{v}_i^{j+1} = \sum_{k=1}^{N_2} \hat{p}_{i,k}^{j+1} \sigma_{i,k}(\mathbf{Z}) = \hat{\mathbf{P}}_{\mathbf{i}}^{j+1} \sigma_{\mathbf{i}}(\mathbf{Z}), \tag{14.60}$$

$$\hat{w}_i^{j+1} = \sum_{k=1}^{N_3} \hat{q}_{i,k}^{j+1} \rho_{i,k}(\mathbf{Z}) = \hat{\mathbf{Q}}_{\mathbf{i}}^{j+1} \rho_{\mathbf{i}}(\mathbf{Z}), \tag{14.61}$$

where $\hat{\mathbf{v}}^{j+1} = [\hat{v}_1^{j+1}, \ldots, \hat{v}_l^{j+1}]$, $\hat{\mathbf{w}}^{j+1} = [\hat{w}_1^{j+1}, \ldots, \hat{w}_q^{j+1}]$. The terms $\phi_{\mathbf{i}}(\mathbf{Z}) = [\phi_{i1}, \ldots, \phi_{iN_{i1}}]$, $\sigma_{\mathbf{i}}(\mathbf{Z}) = [\sigma_{i1}, \ldots, \sigma_{iN_{i2}}]$ and $\rho_{\mathbf{i}}(\mathbf{Z}) = [\rho_{i1}, \ldots, \rho_{iN_3}]$ are basis function vectors, $\hat{\mathbf{C}}^{j+1}$, $\hat{\mathbf{P}}_{\mathbf{i}}^{j+1}$ and $\hat{\mathbf{Q}}_{\mathbf{i}}^{j+1}$ are constant weight vectors and N_1, N_2 and N_3 are the number of neurons. Substituting (14.59)–(14.61) into the off-policy H_∞ Bellman equation (14.55) yields

$$\begin{aligned}
&e^{-\alpha(t_k - t_{k-1})} \hat{\mathbf{C}}^{j+1} \left[\phi(\mathbf{Z}(t_k)) - \phi(\mathbf{Z}(t_{k-1}))\right] = \\
&\int_{t_{k-1}}^{t_k} e^{-\alpha(\tau - t_{k-1})} \Big(\sum_{i=1}^{l} \hat{\mathbf{P}}_i^{j+1} \sigma_{\mathbf{i}}(\mathbf{Z}) \bar{\mathbf{L}}_i (\tanh^{\mathrm{T}}(\hat{\mathbf{v}}_i^j) - \tanh^{\mathrm{T}}(v_i)) \\
&+ 2\gamma^2 \sum_{i=1}^{q} \hat{\mathbf{Q}}_i^{j+1} \rho_{\mathbf{i}}(\mathbf{Z})(w_i - {w_i}^j) - r_a(\mathbf{Z}(t), \hat{\mathbf{v}}^j, \hat{\mathbf{w}}^j)\Big) dt.
\end{aligned} \tag{14.62}$$

By defining $\hat{\mathbf{P}} = \begin{bmatrix} \hat{P}_1 & \ldots & \hat{P}_l \end{bmatrix}$ and $\hat{\mathbf{Q}} = \begin{bmatrix} \hat{Q}_1 & \ldots & \hat{Q}_q \end{bmatrix}$, Eq. (14.62) can be rewritten as

$$\hat{\mathbf{W}}^{\mathrm{T}} \mathbf{h}(t_k) = y(t_k), \tag{14.63}$$

where

$$\hat{\mathbf{W}} = \begin{bmatrix} (\hat{\mathbf{C}}^{j+1})^{\mathrm{T}} & (\hat{\mathbf{P}}^{j+1})^{\mathrm{T}} & (\hat{\mathbf{Q}}^{j+1}\ )^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}},$$

$$\mathbf{h}(t_k)=\begin{bmatrix} e^{-\alpha(t_k-t_{k-1})}\phi(\mathbf{Z}(t_k))-\phi(\mathbf{Z}(t_{k-1})) \\ \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}\sigma_1(\mathbf{Z})\bar{\mathbf{L}}_1(\tanh^{\mathrm{T}}(\hat{v}_1^j)-\tanh^{\mathrm{T}}(v_1))\,d\tau \\ \vdots \\ \int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}\sigma_l(\mathbf{Z})\bar{\mathbf{L}}_l(\tanh^{\mathrm{T}}(\hat{v}_l^j)-\tanh^{\mathrm{T}}(v_l))\,d\tau \\ 2\gamma^2\int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}\rho_1(\mathbf{Z})(w_1-{w_1}^j)\,d\tau \\ \vdots \\ 2\gamma^2\int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}\rho_q(\mathbf{Z})(w_q-{w_q}^j)\,d\tau \end{bmatrix},$$

$$y(t_k)=\int_{t_{k-1}}^{t_k} e^{-\alpha(\tau-t_{k-1})}r_a(\mathbf{Z}(t),\hat{\mathbf{v}}^j,\hat{\mathbf{w}}^j))\,dt.$$

Case 1: Independency in Elements of $\mathbf{L}(\mathbf{u})$

Eq. (14.63) can be solved using the least square method for parameter vector $\hat{\mathbf{W}}$. Then the approximated value function and disturbance input are (14.59) and (14.61), respectively. The control input $\hat{\mathbf{u}}^{j+1}$ is found by determining $\mathbf{L}(\hat{\mathbf{u}}^{j+1})$ based on (14.45) from (14.60). The number of unknown parameters $\hat{\mathbf{W}}$ is $N_1+N_2+N_3$. Then, at least $N \geqslant N_1+N_2+N_3$ data sampled t_1 to t_N should be collect before solving (14.63) in the least square sense,

$$\begin{aligned}\mathbf{Y}&=\begin{bmatrix} y(t_1) & \dots & y(t_N)\end{bmatrix}^T,\\ \mathbf{H}&=\begin{bmatrix} h(t_1) & \dots & h(t_N)\end{bmatrix}.\end{aligned}$$

The least square solution is obtained as

$$\hat{\mathbf{W}}=(\mathbf{H}\mathbf{H}^{\mathrm{T}})^{-1}\mathbf{H}\mathbf{Y}.$$

Case 2: Dependency in Elements of $\mathbf{L}(\mathbf{u})$

If the elements of $\mathbf{L}(\mathbf{u})$ are dependent, one has to solve a constrained nonlinear least square problem to take into account the equality constraints imposed by the dependency of the elements of $\mathbf{L}(\mathbf{u})$. To show this, consider the case of the UAV in Example 1. The following constraints are considered when finding the weights of NNs:

$$\begin{aligned}&\bar{L}_3\tanh(\hat{P}_3^{j+1}\sigma_3(\mathbf{Z}))=(\bar{L}_2\tanh(\hat{P}_2^{j+1}\sigma_2(\mathbf{Z})))^2=\\ &(\bar{L}_4\tanh(\hat{P}_4^{j+1}\sigma_4(\mathbf{Z})))^2+(\bar{L}_5\tanh(\hat{P}_5^{j+1}\sigma_5(\mathbf{Z})))^2.\end{aligned}$$

This constraint is nonlinear in NN weights and thus requires using the nonlinear least square method. In general, (14.58) becomes

$$arg\min_{\hat{\mathbf{W}}}\left\|\hat{\mathbf{W}}\mathbf{H}-\mathbf{Y}\right\|^2 \quad s.t.\ \mathbf{f}(\hat{\mathbf{P}}^{j+1},\sigma_1,\dots,\sigma_l)=0,$$

where the function $\mathbf{f}$ is defined in (14.57) and depends on how the elements of $\mathbf{L}(\mathbf{u})$ and consequently NN weights are related.

REFERENCES

[1] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, vol. 1, MIT Press, Cambridge, 2017.

[2] D.P. Bertsekas, J.N. Tsitsikli, Neuro-Dynamic Programming, Athena Scientific, MA, 1996.

[3] W.B. Powell, Approximate Dynamic Programming, Wiley, Hoboken, NJ, 2007.

[4] F.L. Lewis, D. Liu, Reinforcement Learning and Approximate Dynamic Programming for Feedback Control, Wiley, 2013.

[5] D. Vrabie, K.G. Vamvoudakis, F.L. Lewis, Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles, Institution of Engineering and Technology, 2013.

[6] H. Zhang, L. Cui, X. Zhang, X. Luo, Data-driven robust approximate optimal tracking control for unknown general nonlinear systems using adaptive dynamic programming method, IEEE Transactions on Neural Networks 22 (2011) 2226–2236.

[7] K.G. Vamvoudakis, F.L. Lewis, Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem, Automatica 46 (5) (2010) 878–888.

[8] D. Vrabie, F.L. Lewis, Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems, Neural Networks 22 (3) (2009) 237–246.

[9] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, F.L. Lewis, Adaptive optimal control for continuous-time linear systems based on policy iteration, Automatica 45 (2) (2009) 447–484.

[10] R. Song, W. Xiao, H. Zhang, C. Sun, Adaptive dynamic programming for a class of complex-valued nonlinear systems, IEEE Transactions on Neural Networks and Learning Systems 25 (9) (2014) 1733–1739.

[11] H. Modares, F.L. Lewis, M.B. Naghibi-Sistani, Adaptive optimal control of unknown constrained-input systems using policy iteration and neural networks, IEEE Transactions on Neural Networks and Learning Systems 24 (10) (2013) 1513–1525.

[12] S. Bhasin, R. Kamalapurkar, M. Johnson, K.G. Vamvoudakis, F.L. Lewis, W.E. Dixon, A novel actor-critic-identifier architecture for approximate optimal control of uncertain nonlinear systems, Automatica 49 (1) (2013) 82–92.

[13] T. Bian, Y. Jiang, Z.-P. Jiang, Adaptive dynamic programming and optimal control of nonlinear nonaffine systems, Automatica 50 (10) (2014) 2624–2632.

[14] Y. Jiang, Z.-P. Jiang, Robust adaptive dynamic programming and feedback stabilization of nonlinear systems, IEEE Transactions on Neural Networks and Learning Systems 25 (5) (2014) 882–893.

[15] Y. Jiang, Z.-P. Jiang, Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics, Automatica 48 (10) (2012) 2699–2704.

[16] D. Liu, H. Li, D. Wang, Error bounds of adaptive dynamic programming algorithms for solving undiscounted optimal control problems, IEEE Transactions on Neural Networks and Learning Systems 26 (6) (2015) 1323–1334.

[17] B. Luo, H.-N. Wu, T. Huang, D. Liu, Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design, Automatica 50 (12) (2014) 3281–3290.

[18] B. Luo, H.-N. Wu, H.-X. Li, Adaptive optimal control of highly dissipative nonlinear spatially distributed processes with neuro-dynamic programming, IEEE Transactions on Neural Networks and Learning Systems 26 (4) (2015) 684–696.

[19] M. Abu-Khalaf, F.L. Lewis, J. Huang, Neurodynamic programming and zero-sum games for constrained control systems, IEEE Transactions on Neural Networks and Learning Systems 19 (7) (2008) 1243–1252.

[20] H. Zhang, Q. Wei, D. Liu, An iterative adaptive dynamic programming method for solving a class of nonlinear zero-sum differential games, Automatica 47 (1) (2011) 207–214.

[21] K.G. Vamvoudakis, F.L. Lewis, Online solution of nonlinear two-player zero-sum games using synchronous policy iteration, International Journal of Robust and Nonlinear Control 22 (13) (2012) 1460–1483.

[22] K.G. Vamvoudakis, F.L. Lewis, Online Gaming: Real Time Solution of Nonlinear Two-Player Zero-Sum Games Using Synchronous Policy Iteration, INTECH Open Access Publisher, 2011.

[23] H. Modares, F.L. Lewis, M.-B. Naghibi-Sistani, Online solution of nonquadratic two-player zero-sum games arising in the H_∞ control of constrained input systems, International Journal of Adaptive Control and Signal Processing 28 (3–5) (2014) 232–254.

[24] H. Zhang, C. Qin, B. Jiang, Y. Luo, Online adaptive policy learning algorithm for H_∞ state feedback control of unknown affine nonlinear discrete-time systems, IEEE Transactions on Cybernetics 44 (12) (2014) 2706–2718.

[25] H.-N. Wu, B. Luo, Simultaneous policy update algorithms for learning the solution of linear continuous-time H_∞ state feedback control, Information Sciences 222 (2013) 472–485.

[26] H.-N. Wu, B. Luo, Neural network based online simultaneous policy update algorithm for solving the HJI equation in nonlinear H_∞ control, IEEE Transactions on Neural Networks and Learning Systems 23 (12) (2012) 1884–1895.

[27] B. Luo, H.-N. Wu, Computationally efficient simultaneous policy update algorithm for nonlinear H_∞ state feedback control with Galerkin's method, International Journal of Robust and Nonlinear Control 23 (9) (2013) 991–1012.

[28] D. Vrabie, F.L. Lewis, Adaptive dynamic programming for online solution of a zero-sum differential game, Journal of Control Theory and Applications 9 (3) (2011) 353–360.

[29] H. Li, D. Liu, D. Wang, Integral reinforcement learning for linear continuous-time zero-sum games with completely unknown dynamics, IEEE Transactions on Automation Science and Engineering 11 (3) (2014) 706–714.

[30] B. Luo, H.-N. Wu, T. Huang, Off-policy reinforcement learning for H_∞ control design, IEEE Transactions on Cybernetics 45 (1) (2015) 65–76.

[31] R.A. Howard, Dynamic Programming and Markov Processes, MIT Press, Cambridge, MA, 1960.

[32] K.G. Vamvoudakis, D. Vrabie, F.L. Lewis, Online adaptive algorithm for optimal control with integral reinforcement learning, International Journal of Robust and Nonlinear Control 24 (17) (2014) 2686–2710.

[33] B. Kiumarsi, F.L. Lewis, H. Modares, A. Karimpour, M.-B. Naghibi-Sistani, Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics, Automatica 50 (4) (2014) 1167–1175.

[34] H. Zhang, Q. Wei, Y. Luo, A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy HDP iteration algorithm, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 38 (2008) 937–942.

[35] D. Wang, D. Liu, Q. Wei, Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach, Neurocomputing 78 (2012) 14–22.

[36] T. Dierks, S. Jagannathan, Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics, in: Decision and Control, 2009 Held Jointly with the 2009 28th Chinese Control Conference, CDC/CCC 2009, Proceedings of the 48th IEEE Conference on, 2009, pp. 6750–6755.

[37] T. Dierks, S. Jagannathan, Optimal control of affine nonlinear continuous-time systems, in: American Control Conference, ACC, 2009, pp. 6750–6755.

[38] M.D.S. Aliyu, Nonlinear H_∞ Control, Hamiltonian Systems and Hamilton–Jacobi Equations, CRC Press, 2017.

[39] S. Devasia, D. Chen, B. Paden, Nonlinear inversion-based output tracking, IEEE Transactions on Automatic Control 41 (7) (1996) 930–942.

[40] G.J. Toussaint, T. Basar, F. Bullo, H_∞-optimal tracking control techniques for nonlinear underactuated systems, in: Decision and Control, 2000, Proceedings of the 39th IEEE Conference on, vol. 3, 2000, pp. 2078–2083.

[41] J.A. Ball, P. Kachroo, A.J. Krener, H_∞ tracking control for a class of nonlinear systems, IEEE Transaction on Automatic Control 44 (6) (1999) 1202–1206.

[42] T. Basar, P. Bernard, H_∞ Optimal Control and Related Minimax Design Problems, Birkhäuser, Boston, MA, 1995.

[43] F.L. Lewis, D. Vrabie, V. Syrmos, Optimal Control, 3rd edition, Wiley, 2012.

[44] H. Modares, F.L. Lewis, 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 26 (2015) 2550–2562.

[45] B.L. Stevens, F.L. Lewis, E.N. Johnson, Aircraft Control and Simulation, third edition, Wiley-Blackwell, 2015.

[46] B. Kiumarsi, W. Kang, F.L. Lewis, H_∞ control of non-affine aerial systems using off-policy reinforcement learning, Unmanned Systems 4 (1) (2016) 51–60.