Tracking Control for Linear Discrete-Time Networked Control Systems With Unknown Dynamics and Dropout

Yi Jiang, Student Member, IEEE, Jialu Fan, Member, IEEE, Tianyou Chai, Fellow, IEEE, Frank L. Lewis, Fellow, IEEE, and Jinna Li, Member, IEEE

Abstract—This paper develops a new method for solving the optimal control tracking problem for networked control systems (NCSs), where network-induced dropout can occur and the system dynamics are unknown. First, a novel dropout Smith predictor is designed to predict the current state based on historical data measurements over the communication network. Then, it is shown that the quadratic form of the performance index is preserved even with dropout, and the optimal tracker solution with dropout is given based on a novel dropout generalized algebraic Riccati equation. New algorithms for offline policy iteration (Pl), online Pl, and Q-learning Pl are presented for NCS with dropout. The Q-learning algorithm adaptively learns the optimal control online using data measured over the communication network based on reinforcement learning, including dropout, without requiring any knowledge of the system dynamics. Simulation results are provided to show that the proposed approaches give proper optimal tracking performance for the NCS with unknown dynamics and dropout.

Index Terms—Dropout, networked control system (NCS), Q-learning, reinforcement learning (RL).

I. INTRODUCTION

IN NETWORKED control systems (NCSs), communication networks are employed to exchange the information and control signals (reference input, feedback state, and control input signals) between the sensors, the controllers, and the actuators. With the introduction of the communication networks, the advantages of NCSs are low cost, reduced weight, simple installation and maintenance, and high reliability; therefore, NCSs are widely applied to robot control, motor control, industrial manufacture, vehicles, aircraft, and spacecraft [1]–[7].

Despite their advantages and wide applications, communication networks will introduce errors through unreliable transmission, such as network-induced dropout [5]–[8], network-induced time delay [8]–[10], network noise [6], [11], [12], and quantization problem [13]–[17], which are usually caused by limited bit rate of the communication channels, by the sharing of limited network bandwidth, or by a node waiting to send out a packet via a busy channel. It is known that the occurrence of dropout, time delay, noise, and quantization degrade the stability and the control performance of the closed-loop system. Many researchers have paid attention to the study of the controller design for stabilization and proper performance for the NCSs.

For linear NCS with unreliable transmission, [4]–[9] and [13] use Lyapunov functions to obtain a group of linear matrix inequalities, whose solution gives stabilizing feedback gains. Papers [18]–[21] discuss the transmission control protocol (TCP) and the user datagram protocol (UDP), combine the Kalman filter and stochastic optimal control, give the relationship between the stability and the probability of the dropout, and prove that the optimal performance index has a quadratic form. Paper [22] considers the delay compensations in both the feedback and forward channels in NCSs for networked output tracking control problem. Reference [23] proposes the networked-predictive-control scheme to compensate for the network-induced delay. Paper [17] investigates the problem of event-triggered $H_{\infty}$ control for a networked singular system with both state and input subject to quantizations. For the nonlinear case, paper [24] proposes a model predictive control framework to compensate for the two-channel packet dropout. All these approaches require the knowledge of the system dynamics, and some of these approaches even require the statistical characteristics of the communication networks.

Reinforcement learning (RL) [25]–[34] is a class of machine learning method widely used in optimal control to find an optimal policy for unknown system dynamics. Papers [31] and [35] use Q-learning to solve the tracking and zero-sum game problem with unknown system dynamics, [36] proves the convergence of the value iteration (VI), and [37] utilizes
the measured input and output data to implement policy iteration (PI) and VI. Reference [38] uses actor-critic neural networks to learn the optimal control online for uncertain nonlinear discrete-time (DT) systems.

To address unknown system dynamics NCS, an optimal control approach for linear NCS and a network scheduling protocol in a distributed framework is presented in [39]. Stochastic Q-learning method is used to design the optimal controller in an event-sampled framework. This design considers an ideal communication network without any network imperfections. Paper [40] proposes the stochastic optimal control method using an adaptive estimator and ideas from Q-learning to solve the infinite horizon optimal regulation problem for unknown NCS with time-varying system matrices. Then, paper [41] extends this approach to the nonlinear case.

However, these papers do not consider the influence of network-induced feedback dropout. When the dropout occurs in the feedback loop, no current state can be obtained for PI, then, the optimal control cannot be calculated, and even the unstable control input will be calculated. Thus, this paper develops a new Q-learning approach for the tracking problem in linear DT NCSs with unknown dynamics and dropout. A novel Smith predictor is developed to take advantage of the historical data to estimate the current state of the system. This information is used to achieve Q-learning online when dropout occurs.

The contributions of this paper are as follows.
1) A novel network-induced dropout Smith predictor is developed to predict the current state from historical measured data when network-induced dropout occurs.
2) It is proven that the NCS performance index is still quadratic even with dropout, and the optimal tracking control solution is provided by solving an NCS generalized algebraic Riccati equation (GARE).
3) A Q-learning method is given for using PIs to solve the Q-function online using measured data with completely unknown system dynamics and network-induced dropout.

This paper is organized as follows. In Section II, we formulate the NCS optimal control problem with network-induced dropout. In Section III, we establish a novel network-induced dropout Smith predictor, prove the quadratic form of the NCS optimal control problem, and solve this problem by solving a GARE. In Section IV, we provide an off-line PI and an online iteration to solve the GARE off-line and online. In Section V, we provide the PI to solve Q-function online using measured data with network-induced dropout. Simulation experiments are used to verify the performance of the algorithms.

II. PROBLEM FORMULATION

The objective of this paper is to make the output of a dynamical system track a reference signal. This is known as the tracking problem. In this section, we introduce the linear quadratic tracker (LQT) problem and formulate the network-induced dropout problem, where some portions of signals can be lost due to failures in a communication system.

We formulate the system of interest as the DT linear system having dynamics

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k)$$

(1)

where $x \in \mathbb{R}^{n_x}$ is the state, $u \in \mathbb{R}^{n_u}$ is the input, and $y \in \mathbb{R}^{n_y}$ is the output, respectively. $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, and $C \in \mathbb{R}^{n_y \times n_x}$ are constant matrices.

This is a standard form used to model a system that develops through time in response to control $u(k)$ and generates an output signal $y(k)$. Our objective in the tracker problem is to design a control input $u(k)$ that makes the output $y(k)$ follow a desired reference signal $r(k)$. We take the reference signal as generated by the dynamical system

$$r(k+1) = Fr(k)$$

(2)

where $r \in \mathbb{R}^{n_r}$ and $F \in \mathbb{R}^{n_r \times n_r}$. In the tracker problem, it is desired for the output $y(k)$ of system (1) to track the reference signal $r(k)$.

A networked control system is shown in Fig. 1. Therefore, the state $x(k)$ is measured and passes through a communication network to a controller, which computes $u(k)$. It is assumed that the state information is transmitted with a single packet and that some packets are lost in transmission, known as network-induced dropout. Therefore, the state $x_f(k)$ available to the controller with network-induced dropout is

$$x_f(k) = \delta(k)x(k) + (1 - \delta(k))x_f(k - 1)$$

(3)

where $\delta(k) \sim (0,1)$, $\delta(k) = 0$ means dropout occurs and $\delta(k) = 1$ means a successful transmission. Then, the following assumptions are made.

Assumption 1: The pair $(A,B)$ is stabilizable and the pair $(A,C)$ is observable.

Assumption 2: The maximum number of contiguous feedback packet dropout $\delta(k)$ is $\delta_f \max$, equivalently

$$\delta_f \max \sum_{i=0}^{\delta_f \max} \delta(k - i) > 0.$$  

(4)

This paper assumes that the number of contiguous feedback packet dropout $\delta(k)$ is bounded; when the contiguous feedback packet dropout number is infinite, the control system is equivalent to an open-loop control system.
To solve the tracker problem, construct the augmented system as
\[
X(k+1) = \begin{bmatrix} A & 0 & F \\ r(k) & 0 \end{bmatrix} x(k) + \begin{bmatrix} B \\ 0 \end{bmatrix} u(k) = A_2 X(k) + B_2 u(k)
\]
\[
y(k) = [C \ 0] X(k) \equiv C_2 X(k)
\]
where the augmented state is \( X(k) = \begin{bmatrix} x(k) \\ r(k) \end{bmatrix} \). In the tracker problem, it is desired for the system output \( y(k) \) in (1) to follow the reference signal \( r(k) \) generated by (2). This is accomplished by selecting a suitable control signal \( u(k) \) in (1). To formulate a design problem for choosing \( u(k) \) to make \((r(k) - y(k))\) go to zero, thus achieving tracking, define an LQ problem performance index as
\[
J(k) = \frac{1}{2} \sum_{i=k}^{\infty} \gamma^{i-k} [r(i) - y(i)]^T Q (r(i) - y(i)) + u^T (i) Ru(i)
\]
\[
= \frac{1}{2} \sum_{i=k}^{\infty} \gamma^{i-k} U(i)
\]
\[
= \frac{1}{2} U(k) + \gamma J(k+1)
\]
where \( 0 < \gamma \leq 1 \) is a discount factor. The tracker problem is solved by selecting control \( u(k) \) to minimize this performance index. If the reference generator (2) is asymptotically stable, one may select \( \gamma = 1 \). If (2) is critical stable, e.g., in tracking a unit step, then it is required that \( \gamma < 1 \). In fact, one selects the discount factor so that \((\gamma^0.5 F)\) is stable, which can be proven in Section III.

**Remark 1:** The reference generator (2) is a standard form used in many works. The system is said to be stable if signal \( r(k) \) goes to zero with time. In general, we assume that \( F \) is not stable. If \( F \) is stable, then a trivial control guarantees trajectory following by making \( y(k) \) go to zero. The model (2) can describe many sorts of signals. By selecting \( F \) to have one pole at the origin, we can generate a whole family of unit step reference signals. If \( F \) has nonrepeated poles on the imaginary axis, we can generate a family of sinusoidal reference signals.

## III. SOLUTION TO THE DROPOUT LQT PROBLEM

In this section, we introduce a dropout Smith predictor to predict the current state when network-induced dropout occurs. Using this Smith predictor, we establish that the performance index (6) has a quadratic form. The solution to the dropout LQT problem is given in terms of a GARE. This section develops a dropout LQT Bellman equation, which is used in Section V to use RL to solve the dropout LQT problem online when the dynamics (1) and (2) are unknown.

### A. Network-Induced Dropout Smith Predictor

The solution to the LQ problem requires knowledge of the states \( x(k) \) and \( r(k) \). In this section, we design a dropout Smith predictor that uses the past values of the available measurements with dropout (3) to predict the current state \( X(k) \) in (5).

The feedback information \( x(k) \) is transmitted through the network using a single packet that may succumb to network-induced dropout as in (3). Assume that the network control system communication protocol is the TCP or the UDP. In the TCP case, there is acknowledgment of received packets, while in the UDP case, no feedback is provided on the communication link. But, using either TCP or UDP, the dropout history is known and can be used to improve the controller design.

Now, define the number of contiguous packet dropouts that has occurred prior to step \( k \) as \( \delta_f(k) \). If \( x(k) \) is transmitted successfully according to (3), we define \( \delta_f(k) = 0 \). Otherwise, \( 0 \leq \delta_f(k) \leq \delta_{f,\text{max}} \). Then, the most recent data available through the network at time \( k \) are
\[
x_f(k) = x(k - \delta_f(k)).
\]

With this definition of contiguous feedback packet dropout number, the network-induced dropout can be considered as a stochastic bounded time delay. Then, the current state can be predicted by using historical state and control input information as
\[
x(k) = A^\delta_f(k) x(k - \delta_f(k)) + \sum_{i=1}^{\delta_f(k)} A^{i-1} B u(k-i).
\]

Using TCP or UDP, \( \delta_f(k) \) is known. Equation (7) gives the most recent data available through the network at time \( k \) as \( x(k - \delta_f(k)) \). Equation (8) explicitly computes current state \( x(k) \) in terms of this most recent data and the controls \( u(k-i) \) that have been applied, since time \( k - \delta_f(k) \).

Now, a dropout Smith predictor [42] using the measurable state can be constructed. Define the available measurement vector at time \( k \) as
\[
z(k) = \begin{bmatrix} x_f^T(k) \\ \vdots \\ x_f^T(k) \end{bmatrix}_{(\delta_{f,\text{max}}+1)} U(k-1) u^T(k-\delta_{f,\text{max}}) \ldots u^T(k) r^T(k) \end{bmatrix}^T.
\]

Thus, the predictive state is
\[
X(k) = MN(\delta_f(k)) z(k)
\]
where
\[
M = \begin{bmatrix} I & A & \cdots & A^{\delta_{f,\text{max}}} & B & AB & \cdots & A^{\delta_{f,\text{max}}-1} B & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & I \end{bmatrix}
\]
and \( N(\delta_f(k)) \) is a \([n_x \times (\delta_{f,\text{max}}+1) + n_u \times \delta_{f,\text{max}} + n_r] \times [n_x \times (\delta_{f,\text{max}}+1) + n_u \times \delta_{f,\text{max}} + n_r] \) matrix depending on \( \delta_f(k) \) defined as follows. Define \( \tilde{z}(k) = N(\delta_f(k)) z(k) \) that when \( \delta_f(k) = 0,1,\cdots, \delta_{f,\text{max}}, \tilde{z}(k) \) is calculated,
respectively, as
\[
\tilde{z}(k) = \begin{bmatrix} x_T^T(k) & 0 & \cdots & 0 & 0 & \cdots & 0 & r^T(k) \end{bmatrix}^T \quad \delta_{f_\alpha}(k) = 0
\]
(12)
\[
\tilde{z}(k) = \begin{bmatrix} 0 & x_T^T(k) & \cdots & 0 \end{bmatrix} \quad \delta_{f_\alpha}(k) = \delta_{f_{\max}}
\]
(13)
\[
\tilde{z}(k) = \begin{bmatrix} 0 & 0 & \cdots & x_T^T(k) \end{bmatrix} \quad \delta_{f_\alpha}(k) = \delta_{f_{\max}}
\]
(14)

Note that \( \tilde{z}(k) \) is known at time \( k \). Then, the predictive state \( (5) \) can be described as
(15)

\[ X(k) = M \tilde{z}(k). \]

B. Quadratic Form of the Dropout LQT Problem

With augmented system \( (5) \), the network-induced dropout LQT problem performance index is
(16)
\[
J(k) = \frac{1}{2} \sum_{i=k}^{\infty} \gamma^i \left[ X^T(i)Q_1X(i) + u^T(i)Ru(i) \right]
\]
where \( Q_1 = C_1^TQ_1C_1 \) with \( C_1 = [C - I] \). Define the control input as
\[
u(k) = K_1x(k) + K_2\gamma(k)
\]
(17)
\[
= KX(k) = KM\tilde{z}(k) \equiv \tilde{K}\tilde{z}(k).\]

**Theorem 1:** Take any stabilizing input \( (17) \) and select discount factor \( \gamma \), such that \( (\gamma^{0.5}F) \) is stable. Then, the dropout LQT problem performance index has a quadratic form
(18)
\[
J(k) = \frac{1}{2} X^T(k)PX(k) = \frac{1}{2} \tilde{z}^T(k)\tilde{P}\tilde{z}(k)
\]
for some matrix \( P = \tilde{P}^T > 0 \), where \( \tilde{P} = \tilde{P}^T = M^TPM > 0 \).

**Proof:** Refer to [35], for any fixed stabilizing policy \( (17) \); then, the dropout LQT problem performance index has a quadratic form depending on \( X(k) \) as
(19)
\[
J(k) = \frac{1}{2} X^T(k)PX(k)
\]
where \( P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \), the definitions of \( P_{11}, P_{12}, P_{21} \), and \( P_{22} \) are [35]
\[
P_{11} = \sum_{i=0}^{\infty} \gamma^i \left[ (A_i^T)^T(C^TQ + K_i^T(RK_i))A_i \right]
\]
\[
P_{12} = \sum_{i=0}^{\infty} \gamma^i \left[ (A_i^T)^T(-C^TQ + K_i^T(RK_i))F_i \right.
\]
\[
+ (A_i^T)^T(C^TQ + K_i^T(RK_i))T
\]
\[
P_{21} = P_{12}^T
\]
\[
P_{22} = \sum_{i=0}^{\infty} \gamma^i \left[ (F_i)^T(C^TQ + K_i^T(RK_i))T \right.
\]
\[
+ (F_i)^T(-C^TQ + K_i^T(RK_i))T
\]
\[
+ (F_i)^T(Q + K_i^T(RK_i))F_i^T
\]
where \( A_e = A + B\gamma(k) \) and \( T = \sum_{n=0}^{i=0} A_i^T(n-1)B\gamma(k)F_i^T \), when \( (\gamma^{0.5}F) \) is stable, \( P_{22} \) is bounded. Replace \( X(k) \) by \( \tilde{z}(k) \), then one obtains
(20)
\[
J(k) = \frac{1}{2} X^T(k)PX(k)
\]
\[
= \frac{1}{2} \tilde{z}^T(k)M^TPM\tilde{z}(k)
\]
\[
= \frac{1}{2} \tilde{z}^T(k)\tilde{P}\tilde{z}(k)
\]
where \( \tilde{P} = M^TPM \).
The proof is completed.

C. Solution to the Dropout LQT Problem

In this section, we derive the Bellman equation for the dropout LQT, and a dropout LQT GARE is given to obtain the solution of the dropout LQT.

Introducing the quadratic form of the performance index \( (18) \) into \( (6) \), we obtain the LQT Bellman equation
(21)
\[
X^T(k)PX(k) = X^T(k)Q_1X(k) + u^T(k)Ru(k)
\]
\[
+ \gamma X^T(k + 1)PX(k + 1).\]

Replacing \( X(k) \) by \( \tilde{z}(k) \), the dropout LQT Bellman equation is
(22)
\[
\tilde{z}^T(k)\tilde{P}\tilde{z}(k) = \tilde{z}^T(k)M^TPM\tilde{z}(k) + u^T(k)Ru(k)
\]
\[
+ \gamma \tilde{z}^T(k + 1)\tilde{P}\tilde{z}(k + 1).\]

Define the LQT Hamiltonian as
(23)
\[
H(k) = \tilde{z}^T(k)M^TPM\tilde{z}(k) + u^T(k)Ru(k)
\]
\[
+ \gamma \tilde{z}^T(k + 1)\tilde{P}\tilde{z}(k + 1) - \tilde{z}^T(k)\tilde{P}\tilde{z}(k).
\]

or
(24)
\[
H(k) = \tilde{z}^T(k)M^TPM\tilde{z}(k) + u^T(k)Ru(k)
\]
\[
+ \gamma J(k + 1) - J(k).\]

Theorem 2 provides the solution to the dropout LQT problem.

**Theorem 2:** The optimal control input of the dropout LQT problem is
(25)
\[
u(k) = KM\tilde{z}(k) = \tilde{K}\tilde{z}(k)
\]

where
(26)
\[
\tilde{K} = (R + \gamma B_2^TPB_2)^{-1}\gamma B_2^TPA_2M
\]
or
(27)
\[
\tilde{K} = (R + \gamma B_2^TM^T\tilde{P}\tilde{M}*B_2)^{-1}\gamma B_2^TM^T\tilde{P}\tilde{M}^*A_2M
\]
where $M^* = M^T (MM^T)^{-1}$ and $\tilde{P}$ satisfies the dropout LQT GARE

$$
M^T Q_1 M - \tilde{P} + \gamma M^T A_2^T M^{*T} \tilde{P} M^* A_2 M - \gamma^2 M^T A_2^T M^{*T} \tilde{P} M^* B_2 (R + \gamma B_2^T M^{*T} \tilde{P} M^* B_2)^{-1} B_2^T M^{*T} \tilde{P} M^* A_2 M = 0
$$

or equivalent to the following LQT ARE:

$$
Q_1 - P + \gamma A_2^T P A_2 - \gamma^2 A_2^T P B_2 \times (R + \gamma B_2^T P B_2)^{-1} B_2^T P A_2 = 0.
$$

Proof: The stationary condition of the LQT Bellman equation is

$$
\frac{\partial H(k)}{\partial u(k)} = 2R u(k) + \gamma \frac{\partial \tilde{z}(k+1)}{\partial u(k)} \frac{\partial J(\tilde{z}(k+1))}{\partial \tilde{z}(k+1)} = 0.
$$

Using (5) and (15), one has

$$
X(k + 1) = M\tilde{z}(k + 1) = A_2 X(k) + B_2 u(k).
$$

Therefore

$$
\tilde{z}(k + 1) = M^* A_2 M\tilde{z}(k) + M^* B_2 u(k)
$$

where $M^* = M^T (MM^T)^{-1}$ is the right reverse of $M$, and thus, $P = M^{*T} \tilde{P} M^*$. The stationary condition of the dropout LQT Bellman equation is

$$
\frac{\partial H(k)}{\partial u(k)} = 2R u(k) + \gamma \frac{\partial \tilde{z}(k+1)}{\partial u(k)} \frac{\partial J(\tilde{z}(k+1))}{\partial \tilde{z}(k+1)} = 2R u(k) + 2\gamma B_2^T M^{*T} \tilde{P} \tilde{z}(k+1) = 0.
$$

Then, the optimal input is

$$
u(k) = -(R + \gamma B_2^T M^{*T} \tilde{P} M^* B_2)^{-1} \gamma B_2^T M^{*T} \tilde{P} M^* A_2 M\tilde{z}(k).
$$

Substituting (5) and (34) in the Bellman equation (22) results in the dropout LQT GARE (28)

$$
M^T Q_1 M - \tilde{P} + \gamma M^T A_2^T M^{*T} \tilde{P} M^* A_2 M - \gamma^2 M^T A_2^T M^{*T} \tilde{P} M^* B_2 (R + \gamma B_2^T M^{*T} \tilde{P} M^* B_2)^{-1} B_2^T M^{*T} \tilde{P} M^* A_2 M = 0.
$$

It is direct to show that this is equivalent to (29).

Theorem 3 shows that the stability and the optimality of the solution by solving dropout LQT GARE.

Theorem 3: Apply the control (25) to the system (5) with network-induced dropout. Select discount factor $\gamma$, such that $(\gamma^{0.5} F)$ is stable, and

$$0 < (P_{11} - C^T QC)(P_{11} + G)^{-1} < \gamma^2 I$$

where $G = P_{11} B(R + B^T P_{11} B)^{-1} R(R + B^T P_{11} B)^{-1} B^T P_{11}$, and then, the closed-loop system (5) is stable. Moreover, this control is optimal and it minimizes the performance index (16).

Proof: The closed-loop system with any fixed stabilizing policy (17) will be

$$x(k + 1) = (A + B K_1)x(k) + B K_2 r(k) = A_c x(k) + B_c r(k).
$$

If the eigenvalues of the closed-loop system $A_c$ are in the unit circle, then the stability of closed-loop system holds.

Inspired by [43] and [44], assume that $\lambda$ is an eigenvalue of the closed-loop system dynamics $A_c$, so that $A_c x_i = \lambda x_i$, where $x_i$ is the feature vector. Then, multiplying the left-hand and right-hand sides of the following closed-loop dynamics LQT ARE by $x_i^T$ and $x_i$:

$$C^T QC - P_{11} + \gamma A_2^T P_{11} A_c + K_1^T R K_1 = 0$$

Then, the optimal input is

$$u(k) = -(R + \gamma B_2^T M^{*T} \tilde{P} M^* B_2)^{-1} \gamma B_2^T M^{*T} \tilde{P} M^* A_2 M\tilde{z}(k).
$$

Subtracting $\gamma^{k+1} J(k + 1)$ from both sides of this equation yields

$$\gamma^{k+1} J(k + 1) - \gamma^k J(k) = -\gamma^k U(k).
$$

Sum both sides of (45) from $k$ to $\infty$ to obtain

$$\gamma^\infty J(\infty) - \gamma^k J(k) = -\sum_{i=k}^{\infty} \gamma^i U(i).
$$

Since $\gamma^\infty J(\infty) = 0$, thus

$$\gamma^k J(k) = \sum_{i=k}^{\infty} \gamma^i U(i).
$$
Sum the left-hand side of (45) from \(k\) to \(\infty\) to obtain
\[
\gamma^k J(k) = \sum_{i=k}^{\infty} [\gamma^i J(i) - \gamma^{i+1} J(i+1)]
\]
\[
= \frac{1}{2} \sum_{i=k}^{\infty} [\gamma^i \tilde{z}^T(i) \tilde{P} \tilde{z}(i) - \gamma^{i+1} \tilde{z}^T(i+1) \tilde{P} \tilde{z}(i+1)].
\]
(48)

Introduce the dropout LQT GARE (35) to (48), and combine it with (47) to yield
\[
\gamma^k J(k) = \frac{1}{2} \gamma^k \tilde{z}^T(k) \bar{P} \tilde{z}(k) + \frac{1}{2} \sum_{i=k}^{\infty} [u(i) + (R + \gamma B_2^T \bar{P} M^* B_2)^{-1} \gamma B_1^T M^* \bar{P} M^* A_2 M \tilde{z}(i)]^T
\times (R + \gamma B_2^T M^* \bar{P} M^* B_2)
\][u(i) + (R + \gamma B_2^T M^* \bar{P} M^* B_2)^{-1} \gamma B_1^T M^* \bar{P} M^* A_2 M \tilde{z}(i)].
\]
(49)

Since \((R + \gamma B_2^T M^* \bar{P} M^* B_2)\) is positive definite, to minimize the value of (49), the optimal control input satisfies the control policy (25) and (27), the proof of the optimality is complete.

The proof is completed. \(\square\)

**Remark 2:** In this paper, we found a bound for \(\gamma\) to guarantee stability. The bound for \(\gamma\) just shows that we need to pick \(\gamma\) close enough to 1.

**Remark 3:** It is shown that the tracking error is not asymptotically stable, but the stability can be guaranteed; it means that the tracking error is not zero because of using the discount factor in the performance index. However, by choosing a proper discount factor and a large enough positive definite matrix \(Q\) in the performance index, one can obtain the steady tracking error as small as desired.

### IV. REINFORCEMENT LEARNING TO SOLVE THE DROPOUT LQT PROBLEM ONLINE

In Section III, it was seen that the optimal tracker control can be computed by finding the solution to the LQT (29) and then computing the optimal control gain feedback using (25) and (26). In this section, we utilize the dropout LQT Bellman equation (22) to design algorithms based on RL that solve the dropout LQT problem without solving the dropout LQT GARE in Theorem 2. This is accomplished in two steps. First, Bellman equation (22) is used to directly develop an off-line PI algorithm (Algorithm 1) that does not require solution of the LQT GARE (29). However, Algorithm 1 still requires knowledge of the system dynamics (1). As such, it is still an offline algorithm that requires repeated solutions of the Lyapunov equation (50). Therefore, we next derive Algorithm 2, which is an online data-based algorithm that finds the optimal control using data measured in real time.

Using the feedback control (17) and (27) introduces them into the Bellman equation (22). Then, it is seen that the dropout LQT Bellman equation is equivalent to the dropout LQT Lyapunov equation
\[
\tilde{P} = M^T Q_1 M + \bar{K}^T R \bar{K}
+ \gamma (M^* A_2 M + M^* B_2 \bar{K})^T \bar{P} (M^* A_2 M + M^* B_2 \bar{K}).
\]
(50)

The following off-line PI algorithm calculates the optimal control by repeatedly solving this dropout LQT Lyapunov equation instead of solving the dropout LQT GARE.

**Algorithm 1** Off-Line PI for Dropout LQT Solution

**Initiation:** Start with a stabilizing control policy \(\bar{K}^1\). Iterate the following two steps on \(j\) until convergence.

1) **Policy Evaluation:** solve for \(\tilde{P}^{j+1}\) using the dropout LQT Lyapunov equation
\[
\tilde{P}^{j+1} = M^T Q_1 M + (\bar{K}^j)^T R \bar{K}^j
+ \gamma (M^* A_2 M + M^* B_2 \bar{K}^j)^T \bar{P}^{j+1} (M^* A_2 M + M^* B_2 \bar{K}^j)
\]
(51)

2) **Policy Improvement:** Update the policy using (27), that is
\[
\bar{K}^{j+1} = -(R + \gamma B_2^T M^* \tilde{P}^{j+1} M^* B_2)^{-1} \gamma B_1^T M^* \tilde{P}^{j+1} M^* A_2 M
\]
(52)

The convergence proof of Algorithm 1 is similar to the convergence proof in [26], and therefore, it is shown that \(\tilde{P}^{j+1}\) is convergent to the solution \(\tilde{P}^*\) of dropout LQT GARE.

Instead of using dropout LQT Lyapunov equation, the next algorithm shows how to utilize data measured online in real time to implement the policy evaluation.

**Algorithm 2** On-Line PI for Dropout LQT Solution

**Initiation:** Start with a stabilizing control policy \(\bar{K}^1\). Iterate the following three steps on \(j\) until convergence.

1) **Policy Evaluation:** solve for \(\tilde{P}^{j+1}\) using the dropout LQT Bellman equation (22)
\[
\tilde{z}^T(k) M^T \tilde{P}^{j+1} M \tilde{z}(k) = \tilde{z}^T(k) M^T (Q_1 + (\bar{K}^j)^T R \bar{K}^j) M \tilde{z}(k)
+ \gamma \tilde{z}^T(k+1) M^T \tilde{P}^{j+1} M \tilde{z}(k+1)
\]
(53)

2) **Policy Improvement:** Update the policy using (26), that is
\[
\bar{K}^{j+1} = -(R + \gamma B_2^T \tilde{P}^{j+1} B_2)^{-1} \gamma B_1^T \tilde{P}^{j+1} A_2
\]
(54)

3) **Variable Transformation:**
\[
\tilde{P}^{j+1} = M^T P^{j+1} M
\]
(55)
\[
\bar{K}^{j+1} = K^{j+1} M
\]
(56)

Policy evaluation (55) is equivalent to
\[
X^T(k) P^{j+1} X(k) = X^T(k) (Q_1 + (K^j)^T R K^j) X(k)
+ \gamma X^T(k+1) P^{j+1} X(k+1).
\]
(57)
To solve (53) or (57), standard least square (LS) or the recursive LS can be utilized as follows. Matrix $P^{j+1}$ is a symmetric $(n_x+n_r) \times (n_x+n_r)$ matrix with $(n_x+n_r) \times (n_x+n_r+1)/2$ independent elements. Therefore, at least $(n_x+n_r) \times (n_x+n_r+1)/2$ data tuples are required before (53) or (57) can be solved using LS. Using the Kronecker product, one has $a^T W b = (b^T \otimes a^T) \vec{w}(W)$. Define

$$
\eta (k) = \begin{bmatrix}
\zeta^T (k) M^T \otimes \zeta^T (k) M^T \\
-\gamma \zeta^T (k+1) M^T \otimes \zeta^T (k+1) M^T \\
\vdots \\
-\gamma \zeta^T (k+s) M^T \otimes \zeta^T (k+s) M^T \\
X^T (k) (Q_1 + (K^T R K^T) X (k) \\
\vdots \\
X^T (k+s) (Q_1 + (K^T R K^T) X (k+s)
\end{bmatrix}
$$

where $s \geq (n_x+n_r) \times (n_x+n_r+1)/2 - 1$. Then, $P^{j+1}$ can be solved using LS as the equation

$$
\vec{w}(P^{j+1}) = (\eta^T (k) \eta (k))^{-1} \eta^T (k) \chi (k). \tag{58}
$$

Both the off-line PI algorithm (Algorithm 1) and the online PI algorithm (Algorithm 2) require the knowledge of the system dynamics. In Section V, we introduce a Q-learning algorithm to avoid the requirement of the knowledge of the system dynamics.

**Remark 4:** Using the VI algorithm [26] instead of using the PI algorithm, an initial stabilizing policy is not needed.

V. Q-LEARNING TO SOLVE DROPOUT LQT BASED ON RL

In Section IV, it was seen that the optimal tracker control can be computed by Algorithms 1 and 2. However, this requires knowing the system dynamics (5), or equivalently (1). Unfortunately, in many cases of interest in process control and elsewhere, the dynamics are unknown. In practice, time signals, such as $u(k), x_f (k)$, and $r(k)$, can be measured in the closed-loop system. Therefore, in this section, we introduce a Q-leaning algorithm based on RL to solve the dropout LQT online, and this algorithm avoids the requirement of the knowledge of the system dynamics.

A. Q-Function of the Dropout LQT

In this section, we establish the Q-function of the dropout LQT.

Based on the dropout LQT Bellman equation (22), the DT dropout LQT Q-function is defined as

$$
Q(k) = \frac{1}{2} \zeta^T (k) M^T Q_1 M \zeta (k) + \frac{1}{2} u^T (k) R u (k) + \frac{1}{2} \gamma \zeta^T (k+1) \tilde{P} \zeta (k+1). \tag{59}
$$

By using augmented system dynamics (5) and (32), (59) becomes

$$
Q(k) = \frac{1}{2} \zeta^T (k) (M^T Q_1 M \zeta (k) + \frac{1}{2} u^T (k) R u (k) + \frac{1}{2} \gamma \zeta^T (k+1) \tilde{P} \zeta (k+1)) + \frac{1}{2} \gamma (M^* A_2 M \tilde{P} (M^* A_2) + P B_2) u (k).
$$

Applying $(\partial Q(k)/\partial u(k)) = 0$ to (61) yields the optimal control input

$$
u (k) = - H_{au}^{-1} H_{uz} \zeta (k).
$$

B. Reinforcement Learning to Solve Q-Function

Online Using Measure Data

To solve for the Q-function online, define the state

$$
Z(k) = \begin{bmatrix}
\zeta (k) \\
u (k)
\end{bmatrix}.
$$

The Q-function satisfies the dropout LQT Bellman equation

$$
Q(k) = \frac{1}{2} \zeta^T (k) M^T Q_1 M \zeta (k) + \frac{1}{2} u^T (k) R u (k) + \gamma Q(k+1).
$$

With state (63), the Q-function Bellman equation is

$$
Z^T (k) H Z (k) = \zeta^T (k) M^T Q_1 M \zeta (k) + u^T (k) R u (k) + \gamma Z^T (k+1) H Z (k+1).
$$

Define

$$
\bar{z} (k) = \begin{bmatrix}
\zeta (k) \\
r (k)
\end{bmatrix}
$$

$$
M = \begin{bmatrix}
I & \bar{M} & 0 \\
0 & 0 & I
\end{bmatrix}
$$

where $\bar{z}_1 (k)$ is the element of $\bar{z} (k)$ form row 1 to $n_x$. Clearly, $\bar{z}_1 (k) = x (k)$ when $\delta (k) = 1$ and $\bar{z}_1 (k) = 0$ when $\delta (k) = 0$. Define $\bar{z}_2 (k)$ as the elements of $\bar{z} (k)$ without $\bar{z}_1 (k)$ and $r (k)$.
Thus, one obtains
\[
\begin{align*}
\bar{z}(k) & = z^T(k) C T Q C \bar{z}(k) + r^T(k) Q r(k) \\
& \quad + \bar{z}_2^T(k) \bar{M}^T C T Q C \bar{M} \bar{z}_2(k) \\
& \quad - 2r^T(k) Q C \bar{z}_1(k) - 2r^T(k) Q C \bar{M} \bar{z}_2(k).
\end{align*}
\] (68)

Using the Kronecker product, define \( \phi(k) = Z^T(k) \otimes Z^T(k) \), \( \sigma(k) = \bar{z}_2^T(k) \otimes \bar{z}_2^T(k) \), and \( \vartheta(k) = \bar{z}_2^T(k) \otimes r^T(k) \). Then, the Q-function Bellman equation is
\[
\phi(k) \text{vec}(H) = \bar{z}^T(k) C T Q C \bar{z}(k) + r^T(k) Q r(k) \\
- 2r^T(k) Q C \bar{z}_1(k) + u^T(k) R u(k) \\
+ \sigma(k) \text{vec}(\bar{M}^T C T Q C M) \\
- 2\vartheta(k) \text{vec}(Q C M) + \gamma \vartheta(k) + 1) \text{vec}(H).
\] (69)

or equivalently
\[
[\phi(k) - \gamma \vartheta(k + 1), -\sigma(k), 2\vartheta(k)] \begin{bmatrix} \text{vec}(H) \\ \text{vec}(\bar{M}^T C T Q C M) \\ \text{vec}(Q C M) \end{bmatrix}
= u^T(k) R u(k) + \bar{z}_1^T(k) C T Q C \bar{z}(k) + r^T(k) Q r(k) \\
- 2r^T(k) Q C \bar{z}_1(k).
\] (70)

Define
\[
\xi(k) = \begin{bmatrix} \\
\phi(k) - \gamma \vartheta(k + 1), -\sigma(k), 2\vartheta(k) \\
\alpha(k + s) - \gamma \vartheta(k + s + 1), -\sigma(k + s), 2\vartheta(k + s) \\
\end{bmatrix}
= \xi(k) N
\] (71)

\[
\zeta(k) = \begin{bmatrix}
\bar{z}_1^T(k) C T Q C \bar{z}(k) + r^T(k) Q r(k) - 2r^T(k) Q C \bar{z}_1(k) \\
\bar{z}_1^T(k + s) C T Q C \bar{z}(k + s) + r^T(k + s) Q r(k + s) - 2r^T(k + s) Q C \bar{z}_1(k + s)
\end{bmatrix}
\] (72)

where \( N \) is the elementary column transformation matrix. Scalar \( s \) is an integral number that depends on the maximum number \( \delta_f \text{max} \) of contiguous feedback packet dropout \( \delta(k) \). Then, one has the rank condition
\[
\text{rank}([\xi^T(k) \zeta(k)]) = S_r
\] (73)

where \( \text{rank}([\cdot]) \) respects the rank of the matrix, and
\[
S_r = \sum_{i=0}^{\delta_f \text{max}} (n_x + n_r + n_u + i \times n_u) \\
\times (n_x + n_r + n_u + i \times n_u + 1)/2 \\
- \delta_f \text{max} \times (n_v + n_u) \times (n_v + n_u + 1)/2 \\
+ (n_x + \delta_f \text{max} \times n_u) \times (n_x + \delta_f \text{max} \times n_u + 1)/2 \\
+ (n_x + \delta_f \text{max} \times n_u) \times n_r.
\]

The Q-function Bellman equation becomes
\[
\bar{z}(k) \begin{bmatrix} \text{vec}(H) \\ \text{vec}(\bar{M}^T C T Q C M) \end{bmatrix} = \zeta(k)
\] (74)

or equivalently
\[
\bar{z}(k) \text{vec}(\bar{H}) = \zeta(k)
\] (75)

where
\[
\begin{bmatrix} \text{vec}(\bar{H}) \\ \text{vec}(\bar{M}^T C T Q C M) \end{bmatrix} = N^{-1} \begin{bmatrix} \text{vec}(H) \\ \text{vec}(\bar{M}^T C T Q C M) \end{bmatrix}
\]. Finally, one obtains
\[
\text{vec}(\bar{H}) = (\zeta^T(k) \zeta(k))^{-1} \zeta^T(k) \zeta(k).
\] (76)

Now, the following algorithm can be written.

**Algorithm 3** PI to Solve Q-Function Online Using Measure Data

**Initiation:** Start with a stabilizing control policy \( \bar{K}^1 \). Iterate the following two steps on \( j \) until convergence.

1) **Policy Evaluation:** solve for \( \bar{H}^{j+1} \) using the LS equation (70)
\[
[\phi(k) - \gamma \vartheta(k + 1), -\sigma(k), 2\vartheta(k)] N^{-1} \begin{bmatrix} \text{vec}(H) \\ \text{vec}(\bar{M}^T C T Q C M) \end{bmatrix}^{j+1}
= (u^T(k))^j R(u(k))^j + \bar{z}_1^T(k) C T Q C \bar{z}(k) + r^T(k) Q r(k) \\
- 2r^T(k) Q C \bar{z}_1(k).
\] (77)

2) **Policy Improvement:** Update the policy using (62), that is
\[
u^{j+1}(k) = -(H_{uu}^{-1})^j H_{u\bar{z}}^{j+1} \zeta(k)
\] (78)

The convergence of Algorithm 3 can be proven as in [45]. Note that, PI using Q-function is performed online and can be implemented without requiring any knowledge of the augmented system dynamics.

**Remark 5:** PI-based adaptive optimal control schemes require a persistent excitation (PE) condition [26] to ensure the sufficient exploration of the state space. If the state almost converges to the desired position and becomes stationary, the PE is no longer satisfied. A white noise signal can be added to the control input to ensure PE qualitatively.

**VI. SIMULATION EXPERIMENTS**

In this section, some simulation experiments are used to verify the effectiveness of the proposed method. We verify that Algorithms 1–3 give the same results.

Consider the following linear DT system:
\[
\begin{align*}
x(k + 1) &= \begin{bmatrix} -1 \\ 2.2 \\ 1.7 \end{bmatrix} x(k) + \begin{bmatrix} 2 \\ 1.6 \end{bmatrix} u(k) \\
y(k) &= \begin{bmatrix} 1 \\ 2 \end{bmatrix} x(k).
\end{align*}
\] (79)

Then, \( n_x = 2, n_u = 1, \) and \( n_y = n_r = 1, \) respectively. The open-loop poles are \(-2.1445\) and \(2.8445\), so the system is unstable. The parameters in performance index (6) are selected as \( Q = 6, R = 1, \) and \( \gamma = 0.8, \) and the reference trajectory is given by
\[
r(k + 1) = -r(k)
\] (80)

with \( r(1) = 5 \) and \( F = -1. \)
The maximum number of contiguous feedback packet dropout is \( \delta_{f_{\text{max}}} = 1 \). Therefore, the dropout Smith predictor matrix is

\[ M = \begin{bmatrix} I & A & B & 0 \\ 0 & 0 & 0 & I \end{bmatrix}. \]  

(81)

The optimal matrices \( P^* \) and \( \tilde{P}^* \) by solving the dropout LQT AREs (28) and (29) are as follows, (82) and (83) shown at the bottom of this page.

Thus, the optimal feedback gains \( K^* \) and \( \tilde{K}^* \) are

\[ K^* = [0.1898 \quad -1.0085 \quad 0.0640] \]  

(84)

\[ \tilde{K}^* = K^* M = [0.1898 \quad -1.0085 \quad -2.4086 \\ -1.3348 \quad -1.2340 \quad 0.0640]. \]  

(85)

A. Off-Line Policy Iteration Simulation Experiments

In this section, Algorithm 1 is used to calculate the optimal matrix \( \tilde{P} \) and the optimal feedback gain \( \tilde{K} \). The initial stabilizing control policy \( \tilde{K}^1 \) is selected as

\[ \tilde{K}^1 = [0.2 \quad -1.1 \quad -2.62 \quad -1.47 \quad -1.36 \quad 0.05]. \]  

(86)

All the initial elements of the \( \tilde{P}^{j+1} \) are 0. Fig. 2 shows the error between \( \tilde{P}^{j+1} \) and the optimal matrix \( \tilde{P} \) during learning. At two iterations, Algorithm 1 is applied, and after four iterations, \( \tilde{P}^{j+1} \) is calculated as (87), shown at the bottom of this page, and the feedback control gain is

\[ \tilde{K}^{j+1} = [0.1898 \quad -1.0085 \quad -2.4086 \\ -1.3348 \quad -1.2340 \quad 0.0640]. \]  

(88)

Fig. 3 shows the convergence of the \( \tilde{P}^{j+1} \) matrix parameters to their optimal values for off-line PI algorithm (Algorithm 1).

B. Online Policy Iteration Simulation Experiments

In this section, the online PI algorithm (Algorithm 2) is applied to system (79). The initial stabilizing control policy \( \tilde{K}^1 \) is also selected as (86), all the initial elements of \( \tilde{P}^{j+1} \) are 0, and the initial state is \( x(1) = [5, -5]^T \). Note that, Algorithm 2 starts at control step 10, and after each 6 control steps, \( \tilde{P}^{j+1} \) updates one time. After three iterations, \( P^{j+1} \) is calculated as

\[ P^{j+1} = \begin{bmatrix} 133.3840 & 16.0531 & 31.1402 \\ 16.0531 & 25.1604 & -10.8271 \\ 31.1402 & -10.8271 & 18.4825 \end{bmatrix} \]  

(82)


(83)


(87)
Fig. 4. Evaluation of the output, the input, and the reference trajectory for off-line PI algorithm (Algorithm 1).

Fig. 5. Error between $P^{j+1}$ and the optimal matrix $P^*$ during learning.

Fig. 6. Convergence of the $P$ matrix parameters to their optimal values for online PI algorithm (Algorithm 2).

Fig. 7. Evaluation of the output, the input, and the reference trajectory for online PI algorithm (Algorithm 2).

Fig. 8. Evaluation of the output, the input, and the reference trajectory for Q-learning PI algorithm (Algorithm 3).

and the feedback control gain is calculated as

$$K^{j+1} = [0.1899 \ -1.0085 \ 0.0640].$$

Fig. 5 shows the error norm between $P^{j+1}$ and the optimal matrix $P^*$ during learning. Fig. 6 shows the convergence of the $P$ matrix parameters to its optimal values for online PI algorithm (Algorithm 2). Fig. 7 shows the evaluation of the output, the input, and the reference trajectory for online PI algorithm (Algorithm 2). It is seen that the online algorithm (Algorithm 2) converges very close to the optimal controller and the results given by Algorithm 1.

C. Policy Iteration to Solve Q-Function Online Simulation Experiments

In this section, the online PI to solve Q-function in Algorithm 3 is applied to system (79).

In Q-function (60), the optimal matrix $\tilde{H}^*$ is as (91), shown at the bottom of this page, thus

$$\tilde{H}^*_{uz} = 10^3 \times \begin{bmatrix} -0.1066 & 0.5663 & 1.3525 \\ 0.7496 & 0.6929 & -0.0360 \end{bmatrix}$$

$$\tilde{H}^*_{uu} = 561.5492.$$

To guarantee the excitation condition (PE), a white noise signal is added to the control input. The initial stabilizing control policy $\tilde{K}^1$ is also selected as (86). All the initial

$$\tilde{H}^* = 10^3 \times \begin{bmatrix} 0.1536 & -0.0915 & -0.3548 \\ -0.0915 & 0.5963 & 1.4034 \\ -0.3548 & 1.4034 & 3.4423 \end{bmatrix}$$

$$\tilde{H}^*_{uz} = 10^3 \times \begin{bmatrix} 0.1518 & 0.1609 & 0.0380 & -0.1066 \\ -0.0915 & 0.5963 & 1.4034 & 0.8308 & 0.7712 & -0.0471 & 0.5663 \\ -0.3548 & 1.4034 & 3.4423 & 1.6761 & 1.5357 & -0.1416 & 1.3525 \\ 0.1518 & 0.8308 & 1.6761 & 1.7159 & 1.6329 & -0.0041 & 0.7496 \\ 0.1609 & 0.7712 & 1.5357 & 1.6329 & 1.5557 & 0.0006 & 0.6929 \\ 0.0380 & -0.0471 & -0.1416 & -0.0041 & 0.0006 & 0.0208 & -0.0360 \\ -0.1066 & 0.5663 & 1.3525 & 0.7496 & 0.6929 & -0.0360 & 0.5615 \end{bmatrix}$$
elements of $\bar{H}_u^{j+1}$ and $\bar{H}_{au}^{j+1}$ are 0, and the initial state is $x(1) = [5, -5]^T$. $\bar{H}_u^{j+1}$ and $\bar{H}_{au}^{j+1}$ in Algorithm 3 are updated, which depend on rank condition (73). After three iterations, $\bar{H}_u^{j+1}$ and $\bar{H}_{au}^{j+1}$ are calculated as

$$\bar{H}_u^{j+1} = 103 \times [0.1066 \
0.7496 \
0.5663 \
1.3525 \
0.6929 \
-0.0360]$$

$$\bar{H}_{au}^{j+1} = 561.5486$$

and the feedback control gain is

$$\bar{K}_j^{j+1} = [0.1898 \
-1.0085 \
-2.4086 \
-1.3349 \
-1.2340 \
0.0640]$$

and the feedback control gain is

$$\bar{K}_j^{j+1} = [0.1898 \
-1.0085 \
-2.4086 \
-1.3349 \
-1.2340 \
0.0640]$$

D. Comparison Simulation Experiment

In this section, a comparison simulation experiment is presented. In Algorithm 3, the feedback control gain $K_1$ and the feedforward control gain $K_2$ with Smith predictor are calculated as (96). In the comparison simulation experiment, the feedback control gain $K_1$ is calculated by Algorithm 1, the Smith predictor is the same as in (81). Then, refer to [22], the feedforward control gain $K_2$ is

$$K_2 = \Pi_{22} - K_1 \Pi_{12}$$

where

$$\begin{bmatrix} \Pi_{11} & \Pi_{12} \\ \Pi_{21} & \Pi_{22} \end{bmatrix} = \Psi^T (\Psi \Psi^T)^{-1}, \text{ and } \Psi = \begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix}.$$
the evaluation equation is given by

\[ IAE_y = \frac{1}{k^*} \sum_{k=1}^{k^*} |r(k) - y(k)| \quad (98) \]

\[ MSE_y = \frac{1}{k^*} \sum_{k=1}^{k^*} |r(k) - y(k)|^2. \quad (99) \]

The evaluation indices are shown in Table I. Clearly, the performance in this paper is better than the compared approach; moreover, the approach of this paper does not require the knowledge of system dynamics.

**Remark 6:** The reference trajectory of the recomparison simulation experiment is saltus step, if the reference trajectory is constant, and the evaluation indices of the compared approach are better than Algorithm 3, but the advantage of Algorithm 3 is a model-free approach.

**VII. CONCLUSION**

This paper learns the NCS tracker problem with unknown dynamics and network-induced dropout. The Q-learning PI are proposed, which can learn the optimal control online using measurable data with network dropout without the requirement of knowledge of system dynamics. The simulation results show that the proposed approach gives good tracking performance for the network-based control system with unknown dynamics and dropout.

The future work will focus on how to utilize the input, output, and reference data to implement the Q-learning PI.

**REFERENCES**


Frank L. Lewis (S’70–M’81–SM’86–F’94) received the bachelor’s degree in physics and electrical engineering and the M.S.E.E. degree from Rice University, Houston, TX, USA, the M.S. degree in aeronautical engineering from the University of West Florida, Pensacola, FL, USA, and the Ph.D. degree from the Georgia Institute of Technology, Atlanta, GA, USA.

He is currently a Distinguished Scholar Professor and a Distinguished Teaching Professor with The University of Texas at Arlington, Fort Worth, TX, USA, the Moncrief-ODonnell Chair of the University of Texas at Arlington Research Institute, Fort Worth, TX, USA, the Qian Ren Thousand Talents Professor and the Project 111 Professor with Northeastern University, Shenyang, China, and a Distinguished Visiting Professor with the Nanjing University of Science and Technology, Nanjing, China. He is involved in feedback control, intelligent systems, cooperative control systems, and nonlinear systems. He holds six U.S. patents, and has authored numerous journal special issues, journal papers, and 20 books, including Optimal Control, Aircraft Control, Optimal Estimation, and Robot Manipulator Control, which are used as university textbooks worldwide.

Dr. Lewis is a member of the National Academy of Inventors, a fellow of the International Federation of Automatic Control and the Institute of Measurement and Control, U.K., a Texas Board of Professional Engineer, a U.K. Chartered Engineer, and a founding member of the Board of Governors of the Mediterranean Control Association. He was a recipient of the Fulbright Research Award, the NSF Research Initiation Grant, the Terman Award from the American Society for Engineering Education, the Gabor Award from the International Neural Network Society, the Honeywell Field Engineering Medal from the Institute of Measurement and Control, U.K., the Neural Networks Pioneer Award from the IEEE Computational Intelligence Society, the Outstanding Service Award from the Dallas IEEE Section, and the Texas Regents Outstanding Teaching Award in 2013. He was elected as an Engineer of the Year by the Fort Worth IEEE Section. He was listed in the Fort Worth Business Press Top 200 Leaders in Manufacturing.

Jinna Li (M’12) received the M.S. and Ph.D. degrees from Northeastern University, Shenyang, China, 2006 and 2009, respectively.

From 2009 to 2011, she held the post-doctoral position with the Laboratory of Industrial Control Networks and Systems, Shenyang Institute of Automation, Chinese Academy of Sciences, Beijing, China. From 2014 to 2015, she was a Visiting Scholar granted by the China Scholarship Council with the Energy Research Institute, Nanyang Technological University, Singapore. From 2015 to 2016, she was a Domestic Young Core Visiting Scholar granted by the Ministry of Education of China with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University. Since 2017, she has been a Visiting Scholar with The University of Manchester, Manchester, U.K. She is currently an Associate Professor with the Shenyang University of Chemical Technology, Shenyang. Her current research interests include operational optimal control, reinforcement learning, distributed optimization control, and approximate dynamic programming.