

Received 15 April 2024, accepted 26 April 2024, date of publication 3 May 2024, date of current version 10 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3396782

## RESEARCH ARTICLE

# Adaptive Neural Network Control Framework for Industrial Robot Manipulators

GULAM DASTAGIR KHAN 

Department of Electrical and Computer Engineering, Sultan Qaboos University, Muscat 123, Oman

e-mail: G.khan1@SQU.EDU.OM

This work was supported by Sultan Qaboos University College of Engineering under Grant IG/ENG/ECED/23/02.

**ABSTRACT** Controlling closed architecture industrial robot manipulators poses significant challenges due to limited access to inner controller configurations and specific control gain structures. The absence of open torque or voltage interfaces further compounds these difficulties. Consequently, traditional methods such as the computed-torque approach often prove inadequate when applied to closed architecture robots, widening the gap between advanced control algorithms and practical industrial requirements. In response, this paper introduces a unified framework that utilizes adaptive neural networks to tackle these challenges in controlling closed architecture industrial manipulators. Our approach operates independently of the robot's dynamics, inner controller configuration, and control gain structure. We provide thorough evidence showcasing the boundedness of all control variables. Moreover, the proposed approach is versatile, allowing for the use of a single joint velocity controller across robotic manipulators employing closed control architecture, even under varying conditions. Our strategy streamlines implementation without requiring complex calculations for updating control variables. Experimental results and comparative studies are provided to illustrate the applicability and effectiveness of our proposed control strategy compared to existing approaches.

**INDEX TERMS** Industrial robot, manipulator, neural networks, adaptive control, closed control architecture.


## I. INTRODUCTION

The realm of closed architecture industrial robot control has garnered significant attention in research circles over an extensive period [1], [2], [3], [4], [5], [6], [7], [8], [9]. Among various industrial robotic systems, the robot manipulator stands out as a prominent choice for experimentation, primarily due to its nonlinear Lagrangian dynamic structure [10], [11], [12], [13]. Devising control schemes for industrial robot manipulators is cumbersome due to the absence of open torque or voltage interfaces. Moreover, the inner controller of such robots remains concealed within proprietary domains, rendering it inaccessible for modifications. These two structural factors give rise to an awkward situation, making it challenging to control closed architecture robotic systems.

Traditional methods, such as the computed-torque approach, calculate torques corresponding to desired joint

motions [14], [15], [16], [17]. However, the absence of a torque interface for feed-forward control renders these conventional techniques unsuitable for industrial robots. Consequently, a significant disparity emerges between the sophisticated control algorithms developed in robotics research and the simplistic control methodologies prevalent in industrial robotic applications. This chasm underscores the pressing need to bridge the gap between advanced control strategies pioneered in academic spheres and the pragmatic demands of industrial robotics.

Lange and Hirzinger [1] were the first to introduce a control strategy for industrial robotics. They proposed position-based control for industrial robotic manipulator by seamlessly incorporating it into conventional control systems. This integration enabled precise positioning through cascaded joint controllers. Grotjahn and Heimann [2] further advanced the field with a model-based feedforward control approach, integrating nonlinear precorrection schemes and trajectory correction terms to enhance control precision and adaptability.

The associate editor coordinating the review of this manuscript and approving it for publication was Leimin Wang .

Recent advancements in the control of closed architecture robots have been notable in the works of Wang et al. [3], [7], [8]. In [3], an adaptive-based controller was introduced for closed architecture robot manipulators. This controller relied on the assumption that a second-order linear time-varying system exhibits uniform exponential stability. In [8], the authors proposed an improved controller where the aforementioned stability assumption is eliminated. Additionally, they introduced adaptive controllers capable of accommodating unknown inner controller gains. This significantly streamlines the design and implementation of adaptive controllers in industrial robots and also addresses an important theoretical stability issue. However, in both [3], [8], the inner controllers' structure is assumed to be known by the user but cannot be modified. Moreover, a specific control gain structure, namely diagonality, was assumed in their analysis. Additionally, the incorporation of closed-loop reference dynamics in [8] poses a non-trivial task, requiring intricate calculations for updating control variables. Furthermore, in [8], the design of the proposed controller or command signal varies depending on diverse conditions, lacking a unified approach. In short, both [3], [8] lack a simplified and easy-to-implement unified control strategy that can be applied regardless of the inner control configuration and control gain structure.

Another significant contribution in this field is highlighted in the research conducted by Khan et al. [6], [9]. Their study introduces an innovative neural network-based control approach specifically tailored for industrial robot manipulators operating within uncertain closed architectures and with unknown dynamics. Diverging from prior studies, their work presents a novel approach employing unconstrained control actions, eliminating the necessity of knowledge for inner controller configuration and control gain structure. However, it is worth noting that in the study by Khan et al., particularly in [6], the boundedness of the joint velocity command signal and its integral was not explicitly demonstrated. Moreover, in [9], the neural network appeared to lack the necessary inputs or arguments to adequately capture the desired integral action of the inner controller.

In our research, we introduce a unified framework that harnesses adaptive neural networks to control closed-architecture industrial manipulators. Our proposed approach offers several distinct advantages:

- 1) In contrast to previous works such as [3] and [8], our method operates independently of the robot's dynamics, inner controller configuration, and control gain structure. This unique feature renders our approach model-free and superior in performance.
- 2) Unlike studies such as [6] and [9], we offer comprehensive evidence demonstrating the boundedness of all control variables, ensuring stability. Additionally, the design of the proposed controller enables it to approximate all possible configurations of the inner controller.
- 3) In contrast to prior works like [3], [8], our approach exhibits global applicability. It allows for the

implementation of a single joint velocity controller across various robotic manipulators with closed control architecture, even in diverse operating conditions.

- 4) Unlike the approach presented in [8], we simplify implementation by eliminating the complexity associated with updating control variables, as observed in previous research efforts [8].

## II. PROBLEM FORMULATION

Let  $Y \in \mathbb{R}^n$  denote the output of an unknown nonlinear function, with  $z \in \mathbb{R}^o$  representing its input. We express the objective function  $Y$  using a neural network (NN) as [5]:

$$Y = G\Theta(z) + \hat{E} \quad (1)$$

where  $G \in \mathcal{R}^{n \times n_{p1}}$  stands for an unknown ideal matrix,  $\Theta(z) \in \mathcal{R}^{n_{p1}}$  represents an activation function, and  $\hat{E} \in \mathcal{R}^n$  denotes an approximation error. In this study, we employ a radial basis function (RBF) neural network where the activation function  $\Theta = [\theta_1, \dots, \theta_{n_{p1}}] \in \mathcal{R}^{n_{p1}}$  is defined as [18]

$$\theta_i = \exp\left(-\frac{\|z - \mu_i\|^2}{\rho_i^2}\right), \quad i = 1, \dots, n_{p1} \quad (2)$$

where  $\mu_i = [\mu_{i1}, \dots, \mu_{io}]^T \in \mathcal{R}^o$  is the center of the  $i$ -th neuron, and  $\rho = [\rho_1, \dots, \rho_{n_{p1}}]^T \in \mathcal{R}^{n_{p1}}$  is the distance parameter, with  $\rho_i > 0$ . Therefore, the activation function  $\Theta$ , as defined in (2), is inherently bounded. This neural network architecture proves capable of accurately approximating nonlinear functions with minimal error, and its weights can be adaptively adjusted in real-time, eliminating the need for a prior learning phase. The approximation error can be further minimized by employing a sufficiently large number of neurons within the neural network.

The dynamics of an n-DOF industrial revolute joints robot manipulator can be written as [19]

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = u \quad (3)$$

where  $M(q) \in \mathcal{R}^{n \times n}$  is the inertia matrix which is symmetric and positive definite,  $S(q, \dot{q}) \in \mathcal{R}^{n \times n}$  is the Coriolis and centrifugal matrix such that  $\dot{M}(q) - 2S(q, \dot{q})$  is skew-symmetric,  $g(q) \in \mathcal{R}^n$  is the gravitational torque,  $\dot{q} \in \mathcal{R}^n$  is the joint velocity and  $u \in \mathcal{R}^n$  is the control input.

In industrial robotic manipulators, the inner control loop commonly employs either a Proportional-Derivative (PD) controller with gravity compensation, given by  $u = -K_P(q - q_c) - K_D(\dot{q} - \dot{q}_c) + g(q)$ , or a Proportional-Integral-Derivative (PID) controller without gravity compensation, represented as  $u = -K_P(q - q_c) - K_I(\int_0^t [q(\zeta) - q_c(\zeta)]d\zeta) - K_D(\dot{q} - \dot{q}_c)$  or with gravity compensation as  $u = -K_P(q - q_c) - K_I(\int_0^t [q(\zeta) - q_c(\zeta)]d\zeta) - K_D(\dot{q} - \dot{q}_c) + g(q)$  [10], [13]. Here,  $K_P$ ,  $K_D$ , and  $K_I \in \mathcal{R}^{n \times n}$  represent the inner controller gains, and  $\dot{q}_c \in \mathcal{R}^n$  denotes the user-defined joint velocity commands. It's noteworthy that irrespective of the specific inner controller structure, a term  $K_D(\dot{q} - \dot{q}_c)$  is typically included for stability purposes. Therefore, the generalized

form of the inner controller for uncertain closed architecture industrial robots can be expressed as

$$u = -K(\dot{q} - \dot{q}_c) + \Upsilon \quad (4)$$

where  $K \in \mathcal{R}^{n \times n}$  is the gain of the inner controller. Unlike in [3] and [8], in this study, we make a moderate assumption regarding the positive definiteness of the inner control gain rather than assuming it to be diagonal.  $\Upsilon \in \mathcal{R}^n$  represents the unknown or hidden part of the inner controller. Hence, our strategy remains independent of both inner controller configuration and control gain structure, providing a superior solution. By using (4), the dynamics of the robotic manipulator in (3) can be presented as

$$M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) = -K(\dot{q} - \dot{q}_c) + \Upsilon. \quad (5)$$

To approximate  $\Upsilon$ , we employ neural networks in (1). Thus,  $\Upsilon$  is given as

$$\Upsilon = G_U \Theta_U(q, \dot{q}, \dot{q}_c, \int_0^t [q(\tau) - q_c(\tau)] d\tau) + E_u, \quad (6)$$

where  $G_U \in \mathcal{R}^{n \times n_a}$  represents the ideal constant weight matrix of the neural network used for estimating the uncertain inner loop controller model,  $\Theta_U(q, \dot{q}, \dot{q}_c, \int_0^t [q(\tau) - q_c(\tau)] d\tau) \in \mathcal{R}^{n_a}$  is the activation function vector and  $E_u \in \mathcal{R}^n$  is the approximation error for the uncertain inner loop controller model. In contrast to [9], the neural network (6) possesses adequate inputs or arguments to precisely approximate various inner controller configurations, including integral action of the inner controller.

Using (6), the dynamics of the robotic manipulator in (5) can further be presented as

$$\begin{aligned} M(q)\ddot{q} + S(q, \dot{q})\dot{q} + g(q) &= -K(\dot{q} - \dot{q}_c) \\ &+ G_U \Theta_U(q, \dot{q}, \dot{q}_c, \int_0^t [q(\tau) \\ &- q_c(\tau)] d\tau) + E_u. \end{aligned} \quad (7)$$

The objective is to design the user-defined joint velocity (or position) command signals to enable the closed architecture robot manipulator to track the desired joint positions given by  $q_d \in \mathbb{R}^n$ , such that the tracking error  $\Delta q = q - q_d \rightarrow 0$  as  $t \rightarrow \infty$ . We make the assumption that the desired signals  $q_d$ ,  $\dot{q}_d$ , and  $\ddot{q}_d$  are bounded.

### III. ADAPTIVE NEURAL NETWORK BASED JOINT POSITION TRACKING CONTROLLER

Following the adaptive control approach in [8], we start by defining the vectors  $\lambda$ ,  $q_r^*$  as

$$\lambda = q - q_r \quad (8)$$

$$q_r^* = \dot{q}_r - \gamma(\lambda + \int_0^t \lambda(\tau) d\tau) \quad (9)$$

where  $\gamma$  is the design parameter. The reference dynamics,  $q_r$ , is given as [8]

$$\ddot{q}_r = \ddot{q}_d - \kappa_1 \Delta \dot{q} - \kappa_0 \Delta q + \beta_1 \dot{\lambda} + \beta_2 \lambda + \beta_3 \int_0^t \lambda(\tau) d\tau, \quad (10)$$

where  $\kappa_1, \kappa_0 > 0$  are the design constants.  $\beta_1, \beta_2, \beta_3 \geq 0$  are design constants. We define the sliding vector as

$$s = \dot{q} - \dot{q}_r^* = \dot{\lambda} + \gamma(\lambda + \int_0^t \lambda(\tau) d\tau). \quad (11)$$

By substituting (8) and (11) in (7), we get

$$\begin{aligned} M(q)(\dot{s} + \dot{q}_r^*) + S(q, \dot{q})(s + \dot{q}_r^*) + g(q) \\ = -K(\dot{q} - \dot{q}_c) + G_U \Theta_U(q, \dot{q}, \dot{q}_c, \int_0^t [q(\tau) - q_c(\tau)] d\tau). \end{aligned} \quad (12)$$

Utilizing (9), we can further express (12) as:

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s + M(q)\ddot{q}_r + S(q, \dot{q})\dot{q}_r + g(q) \\ + \left[ -M(q) \frac{d}{dt} (\gamma(\lambda + \int_0^t \lambda(\tau) d\tau)) - S(q, \dot{q})(\gamma(\lambda \right. \\ \left. + \int_0^t \lambda(\tau) d\tau)) \right] \\ = -K(\dot{q} - \dot{q}_c) + G_U \Theta_U(q, \dot{q}, \dot{q}_c, \int_0^t [q(\tau) - q_c(\tau)] d\tau). \end{aligned} \quad (13)$$

We approximate the terms  $-M(q) \frac{d}{dt} (\gamma(\lambda + \int_0^t \lambda(\tau) d\tau)) - S(q, \dot{q})(\gamma(\lambda + \int_0^t \lambda(\tau) d\tau))$  and the dynamic model  $M(q)\ddot{q}_r + S(q, \dot{q})\dot{q}_r + g(q)$  using distinct neural networks as:

$$\begin{aligned} G_\lambda \Theta_\lambda(q, \dot{q}, \lambda, \dot{\lambda}, \int_0^t \lambda(\tau) d\tau) + E_\lambda \\ = -M(q) \frac{d}{dt} (\gamma(\lambda + \int_0^t \lambda(\tau) d\tau)) - S(q, \dot{q})(\gamma(\lambda \\ + \int_0^t \lambda(\tau) d\tau)), \end{aligned} \quad (14)$$

$$\begin{aligned} G_d \Theta_d(q, \dot{q}, \dot{q}_r, \ddot{q}_r) + E_d \\ = M(q)\ddot{q}_r + S(q, \dot{q})\dot{q}_r + g(q) \end{aligned} \quad (15)$$

where  $G_\lambda \in \mathcal{R}^{n \times n_b}$  represents the ideal constant weight matrix of the neural network used for estimating the unknown term  $-M(q) \frac{d}{dt} (\gamma(\lambda + \int_0^t \lambda(\tau) d\tau)) - S(q, \dot{q})(\gamma(\lambda + \int_0^t \lambda(\tau) d\tau))$ .  $\Theta_\lambda(q, \dot{q}, \lambda, \dot{\lambda}, \int_0^t \lambda(\tau) d\tau) \in \mathcal{R}^{n_b}$  is the activation function vector and  $E_\lambda \in \mathcal{R}^n$  is the approximation error. Similarly,  $G_d \in \mathcal{R}^{n \times n_c}$  represents the ideal constant weight matrix of the neural network used for estimating the dynamic model,  $\Theta_d(q, \dot{q}, \dot{q}_r, \ddot{q}_r) \in \mathcal{R}^{n_c}$  is the activation function vector and  $E_d \in \mathcal{R}^n$  is the approximation error for dynamic model. Thus the proposed control scheme is model-free, operating independently of robot dynamics.

Therefore, (13) can be presented as

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s + G_d \Theta_d + E_d + G_\lambda \Theta_\lambda + E_\lambda \\ = -K(\dot{q} - \dot{q}_c) + G_U \Theta_U + E_u. \end{aligned} \quad (16)$$

For convenience, we omit explicitly specifying the parameters of all activation functions in the presentation of neural networks.

Let  $\hat{G}_d$ ,  $\hat{G}_\lambda$ , and  $\hat{G}_U$  represent the estimations of  $G_d$ ,  $G_\lambda$ , and  $G_U$ , respectively. Then by adding and subtracting terms  $\hat{G}_d\Theta_d$ ,  $\hat{G}_\lambda\Theta_\lambda$  and  $\hat{G}_U\Theta_U$  to (16), we get

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s &= -G_d\Theta_d - G_\lambda\Theta_\lambda - K(\dot{q} - \dot{q}_c) + G_U\Theta_U + E \\ &= -G_d\Theta_d + \hat{G}_d\Theta_d - \hat{G}_d\Theta_d - G_\lambda\Theta_\lambda + \hat{G}_\lambda\Theta_\lambda - \hat{G}_\lambda\Theta_\lambda \\ &\quad - K(\dot{q} - \dot{q}_c) + G_U\Theta_U - \hat{G}_U\Theta_U + \hat{G}_U\Theta_U + E \\ &= -\Delta G_d\Theta_d - \hat{G}_d\Theta_d - \Delta G_\lambda\Theta_\lambda - \hat{G}_\lambda\Theta_\lambda - K(\dot{q} - \dot{q}_c) \\ &\quad + \Delta G_U\Theta_U + \hat{G}_U\Theta_U + E \end{aligned} \quad (17)$$

where  $\Delta G_d = G_d - \hat{G}_d$ ,  $\Delta G_\lambda = G_\lambda - \hat{G}_\lambda$ ,  $\Delta G_U = G_U - \hat{G}_U$  and  $E = E_u - E_d - E_\lambda$ .

The joint velocity (or position) command  $\dot{q}_c$  is given as

$$\dot{q}_c = \dot{q}_r + \hat{G}(\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s)), \quad (18)$$

where  $\hat{G} \in \mathcal{R}^{n \times n}$  represents the estimation of  $K^{-1}$  and  $k_g$  is a positive design constant. The control parameter or gain  $k_g > 0$  can be adjusted to zero by utilizing a neural network with a sufficiently large number of neurons. The term  $\hat{G} \times \hat{G}_U\Theta_U$  effectively addresses the uncertainty inherent in the closed-architecture control system of industrial robots. The remaining terms of the command signal (18) ensure the overall stability of the closed-loop system.

The estimated weight matrices are updated by the following neural network based update laws

$$\dot{\hat{G}}_{d[j]} = -\Gamma_d^{-1}\Theta_{d[j]}s \quad (19)$$

$$\dot{\hat{G}}_{U[j]} = -\Gamma_U^{-1}\Theta_{U[j]}s \quad (20)$$

$$\dot{\hat{G}}_{\lambda[j]} = -\Gamma_\lambda^{-1}\Theta_{\lambda[j]}s \quad (21)$$

where  $\hat{G}_{d[j]}$ ,  $\hat{G}_{U[j]}$  and  $\hat{G}_{\lambda[j]}$  denotes the  $[j]th$  column vector of  $\hat{G}_d$ ,  $\hat{G}_U$  and  $\hat{G}_\lambda$ , respectively.  $\Theta_{d[j]}$ ,  $\Theta_{U[j]}$  and  $\Theta_{\lambda[j]}$  denotes the  $[j]th$  element of vectors  $\Theta_d$ ,  $\Theta_U$  and  $\Theta_\lambda$ , respectively.  $\Gamma_d^{-1}$ ,  $\Gamma_U^{-1}$ ,  $\Gamma_\lambda^{-1} \in \mathcal{R}^{n \times n}$  are positive definite matrices.

The update law for  $\hat{G}$  is introduced as

$$\dot{\hat{G}}_{[j]} = -l_j r_{[j]}s \quad (22)$$

where  $\hat{G}_{[j]}$  denotes the  $[j]th$  column vector of  $\hat{G}$ .  $l_j > 0$  is the update gain for the  $[j]th$  column.  $r_{[j]}$  is the  $[j]th$  element of a vector  $r$  defined as  $r = (\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s))$ . Hence, the control methodology is predicated on the notion of formulating  $\dot{q}_c$  to encompass approximations of  $K^{-1}$  and  $\Upsilon$ , alongside the necessary framework of the inner controller to ensure closed-loop stability. This approach facilitates the preservation of the desired inner controller structure upon closure of the system through the cancellation of  $K^{-1}$  and  $\Upsilon$ , consequently ensuring closed-loop stability.

From (17) and (18), we can describe the closed-loop dynamics as

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s &= -\Delta G_d\Theta_d - \hat{G}_d\Theta_d - \Delta G_\lambda\Theta_\lambda - \hat{G}_\lambda\Theta_\lambda - K\dot{q} + \Delta G_U\Theta_U \\ &\quad + \hat{G}_U\Theta_U + K\dot{q}_r + K\hat{G}(\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s)) + E. \end{aligned} \quad (23)$$

The closed-loop dynamics (23), can further be written as

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s &= -\Delta G_d\Theta_d - \hat{G}_d\Theta_d - \Delta G_\lambda\Theta_\lambda - \hat{G}_\lambda\Theta_\lambda - K(\dot{q} - \dot{q}_r) \\ &\quad + K\hat{G}(\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s)) \\ &\quad + \Delta G_U\Theta_U + \hat{G}_U\Theta_U \\ &\quad + k_g \text{sgn}(s) - k_g \text{sgn}(s) + E \\ &= -K\dot{\lambda} - \Delta G_d\Theta_d - \Delta G_\lambda\Theta_\lambda + \Delta G_U\Theta_U - \hat{G}_\lambda\Theta_\lambda + E \\ &\quad - K\Delta\hat{G}(\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s)) - k_g \text{sgn}(s), \end{aligned} \quad (24)$$

where  $\Delta\hat{G} = K^{-1} - \hat{G}$ .

The differential-cascaded system describing the closed-loop dynamics can be presented as

$$\Delta\ddot{q} = -\kappa_1\Delta\dot{q} - \kappa_0\Delta q + \ddot{\lambda} + \beta_1\dot{\lambda} + \beta_2\lambda + \beta_3 \int_0^t \lambda(\tau)d\tau, \quad (25)$$

$$\begin{aligned} M(q)\dot{s} + S(q, \dot{q})s &= -K\dot{\lambda} - \Delta G_d\Theta_d - \Delta G_\lambda\Theta_\lambda - \hat{G}_\lambda\Theta_\lambda + \Delta G_U\Theta_U + E \\ &\quad - K\Delta\hat{G}(\hat{G}_d\Theta_d - \hat{G}_U\Theta_U - k_g \text{sgn}(s)) - k_g \text{sgn}(s). \end{aligned} \quad (26)$$

Consider the Lyapunov function

$$\begin{aligned} V &= \frac{1}{2}s^T M(q)s + \sum_{j=1}^{n_a} \frac{1}{2}\Delta G_{U[j]}^T \Gamma_U^{-1} \Delta G_{U[j]} \\ &\quad + \sum_{j=1}^{n_b} \frac{1}{2}\Delta G_{\lambda[j]}^T \Gamma_\lambda^{-1} \Delta G_{\lambda[j]} + \sum_{j=1}^{n_c} \frac{1}{2}\Delta G_{d[j]}^T \Gamma_d^{-1} \Delta G_{d[j]} \\ &\quad + \frac{1}{2} \sum_{j=1}^n \frac{1}{l_j} \Delta\hat{G}_{[j]}^T K \Delta\hat{G}_{[j]}. \end{aligned} \quad (27)$$

By differentiating the Lyapunov function (27) with respect to time and using the closed loop-dynamics in (24) and update laws (19), (20), (21) and (22), we get

$$\begin{aligned} \dot{V} &= -s^T K\dot{\lambda} - s^T \hat{G}_\lambda\Theta_\lambda - k_g s^T \text{sgn}(s) + s^T E \\ &= -s^T (K\dot{\lambda} + \hat{G}_\lambda\Theta_\lambda) - k_g s^T \text{sgn}(s) + s^T E \\ &= -s^T (K\dot{\lambda} + \hat{G}\hat{G}^{-1}\hat{G}_\lambda\Theta_\lambda) - k_g s^T \text{sgn}(s) + s^T E. \end{aligned} \quad (28)$$

Given that  $\hat{G}$  represents the estimation of  $K^{-1}$ , it follows that  $\hat{G}^{-1}$  represents the estimation of  $K$ . Therefor by using (11),

we get,

$$\begin{aligned}\dot{V} &= -s^T K(\dot{\lambda} + \hat{G}\hat{G}_\lambda\Theta_\lambda) - k_g s^T \text{sgn}(s) + s^T E \\ &= -\left(\dot{\lambda} + \gamma(\lambda + \int_0^t \lambda(\tau)d\tau)\right)^T K\left(\dot{\lambda} + \hat{G}\hat{G}_\lambda\Theta_\lambda\right) \\ &\quad - k_g s^T \text{sgn}(s) + s^T E.\end{aligned}\quad (29)$$

If the gain parameter  $\gamma$  is set to  $\gamma = \hat{G}$ , and the term  $\lambda + \int_0^t \lambda(\tau)d\tau$  is approximated using  $\hat{G}_\lambda\Theta_\lambda$ , and if the control parameter  $k_g$  is chosen to be greater than the upper bound  $b$  of  $E$ , ensuring that the condition  $s^T E - k_g s^T \text{sgn}(s) \leq -(k_g - b)|s|$  is met, then we obtain:

$$\dot{V} = -\left(\dot{\lambda} + \hat{G}\hat{G}_\lambda\Theta_\lambda\right)^T K\left(\dot{\lambda} + \hat{G}\hat{G}_\lambda\Theta_\lambda\right) \leq 0. \quad (30)$$

*Theorem 1:* Let  $\beta_1, \beta_2, \beta_3 \geq 0$ . The gain parameter  $\gamma$  is chosen as  $\gamma = \hat{G}$ , and  $\hat{G}_\lambda\Theta_\lambda$  represents the approximation of the term  $\lambda + \int_0^t \lambda(\tau)d\tau$ . Additionally, the control parameter  $k_g$  is selected as  $k_g > b$ . Then, the joint velocity (or position) command signal in (18) along with adaptive neural network-based update laws in (19)-(21) and adaptive law in (22) along with the reference dynamics given in (10) ensures the convergence of the joint tracking errors, i.e.,  $\Delta q \rightarrow 0$  and  $\Delta \dot{q} \rightarrow 0$  as  $t \rightarrow \infty$ .

*Proof:* From the Lyapunov function described in (27) and its derivative in (30), we have that variables  $s$ ,  $\hat{G}_\lambda$ ,  $\hat{G}_U$ ,  $\hat{G}_d$ , and  $\hat{G}$  are bounded ( $\in \mathcal{L}_\infty$ ).

Moreover, since  $\dot{\lambda} + \hat{G}\hat{G}_\lambda\Theta_\lambda$  is in  $\mathcal{L}_2$  and  $\hat{G}\hat{G}_\lambda\Theta_\lambda$  is in  $\mathcal{L}_\infty$ , we can conclude that  $\lambda \in \mathcal{L}_2 \cap \mathcal{L}_\infty$ ,  $\int_0^t \lambda(\tau)d\tau \in \mathcal{L}_2 \cap \mathcal{L}_\infty$ ,  $\dot{\lambda} \in \mathcal{L}_2$  and  $\lambda \rightarrow 0$  and  $\int_0^t \lambda(\tau)d\tau \rightarrow 0$  as  $t \rightarrow \infty$ .

Considering  $s = \dot{\lambda} + \gamma(\lambda + \int_0^t \lambda(\tau)d\tau)$ , we deduce that  $\dot{\lambda}$  is also bounded. Thus,  $\int_0^t \ddot{\lambda}(\tau)d\tau = \dot{\lambda} - \dot{\lambda}(0)$  also belongs to both  $\mathcal{L}_2$  and  $\mathcal{L}_\infty$ , indicating the convergence and boundedness of the integral.

In the subsystem (25) of the differential-cascaded system, the output  $x = [\Delta q^T, \Delta \dot{q}^T]^T$  is determined by two inputs:

$y_1 = \ddot{\lambda}$  and  $y_2 = \beta_1 \dot{\lambda} + \beta_2 \lambda + \beta_3 \int_0^t \lambda(\tau)d\tau$ . Given that both  $\int_0^t y_1(\tau)d\tau$  and  $y_2$  are bounded, the output associated with  $y_1$  is square-integrable and bounded [20]. Additionally, based on the properties of stable linear systems [21], the outputs linked to  $y_2$  are also square-integrable and bounded. Consequently, the overall output  $x$  is square-integrable and bounded. Thus, from (10), we conclude that  $\ddot{q}_r \in \mathcal{L}_\infty$ . Moreover, from the implications of (10),  $\Delta \dot{q}$  is in  $\mathcal{L}_\infty$ , then  $\dot{q}$  is also in  $\mathcal{L}_\infty$ , leading to  $\dot{q}_r = \dot{q} - \dot{\lambda}$  being in  $\mathcal{L}_\infty$ . Hence, from (9), both  $q_r^*$  and  $\dot{q}_r^*$  are in  $\mathcal{L}_\infty$ .

In this study, following similar methodologies outlined in prior research (e.g., [7], [8]), we demonstrate the boundedness of  $q_c$  and  $\int_0^t q_c(\tau)d\tau$  by leveraging the input-output characteristics of an inverted dynamics of robot manipulator dynamics within a closed architecture. This approach avoids the need to assume uniform exponential stability of a linear time-varying system, as was required in previous studies such as [3]. Specifically, we analyze the inverted dynamics of an industrial manipulator equipped with a PID controller with

gravity compensation setup in the inner loop. This setup is detailed in [8]. The inverted dynamics equation is given by:

$$\begin{aligned}K(\dot{q} - \dot{q}_c) &= -K_P(q - q_c) - K_I \int_0^t [q(\tau) - q_c(\tau)]d\tau \\ &\quad - \frac{d}{dt}[M(q)\dot{q}] + M(q)\dot{q} - S(q, \dot{q})\dot{q} - g(q).\end{aligned}\quad (31)$$

The linear system (31), with outputs as  $-K_P(q - q_c)$  and  $K_I \int_0^t [q(\tau) - q_c(\tau)]d\tau$ , is exponentially stable and strictly proper when inputs  $y_1 = \frac{d}{dt}[M(q)\dot{q}]$ ,  $y_2 = M(q)\dot{q} - S(q, \dot{q})\dot{q} - g(q)$  are zero. Since  $\dot{q} \in \mathcal{L}_\infty$  and  $M(q)$  is symmetric and uniformly positive definite, the inputs  $\int_0^t y_1(\tau)d\tau$ ,  $y_2$  are bounded. The outputs corresponding to  $y_1$  are bounded [20], as are those for  $y_2$  [21]. Consequently,  $\int_0^t [q(\tau) - q_c(\tau)]d\tau$ ,  $q - q_c \in \mathcal{L}_\infty$ , indicating  $q_c \in \mathcal{L}_\infty$ . A closer examination of (26) alongside the properties of the symmetric and positive definite  $M(q)$  reveals that  $\dot{s} \in \mathcal{L}_\infty$ . Consequently, this implies that both  $\ddot{q}$  and  $\Delta \ddot{q}$  are in  $\mathcal{L}_\infty$ . From (31),  $\dot{q} - \dot{q}_c \in \mathcal{L}_\infty$ , and thus  $\dot{q}_c \in \mathcal{L}_\infty$ . Since  $\Delta \dot{q} \in \mathcal{L}_\infty$  and  $\Delta \ddot{q} \in \mathcal{L}_\infty$ ,  $\Delta q$  and  $\Delta \dot{q}$  are uniformly continuous. Therefore,  $\Delta q \rightarrow 0$  and  $\Delta \dot{q} \rightarrow 0$  as  $t \rightarrow \infty$ .  $\square$

*Remark 1:* The adaptive neural network-based design proposed in this study distinguishes itself from that of [8]. Specifically, our design operates independently of both the inner controller configuration and its gain structure. Instead of requiring diagonalization, as in [8], our approach necessitates a moderate assumption of positive definiteness on the controller gain. Moreover, we introduce a universal joint velocity controller that remains consistent across robotic manipulators with closed control architecture, even amidst varying conditions. In contrast, in [8], the joint velocity (or position) command is contingent upon the inner controller structure and prior knowledge of its parameters.

We can further modify the reference dynamics in (10) as

$$\ddot{q}_r = \ddot{q}_d - \kappa_1 \Delta \dot{q} - \kappa_0 \Delta q + \hat{G}_\lambda\Theta_\lambda \quad (32)$$

to eliminate  $\beta_1, \beta_2, \beta_3$  from the controller design. In the work by Wang and Li [8], the reference dynamics described in (25) are utilized to derive the variables  $\lambda, \dot{\lambda}$ , and  $\int_0^t \lambda(\tau)d\tau$ . These variables are then used to compute  $q_r, \dot{q}_r$ , and  $\ddot{q}_r$ , necessitating a non-trivial calculation procedures for updating control variables (see equations (52)-(54) in [8]).

Our approach addresses this complexity by employing a modified neural network-based reference dynamics, as defined in (32). Since the initial estimates of the weights  $\hat{G}_d(0)$ ,  $\hat{G}_U(0)$ , and  $\hat{G}_\lambda(0)$  can be set to zero [18], we can directly compute  $q_r, \dot{q}_r$ , and  $\ddot{q}_r$ . It is important to note that the neural network compensation becomes inactive when the weights are initialized to zero at  $t = 0$ . However, the feedforward term  $\ddot{q}_d$  and the feedback terms  $\Delta \dot{q}, \Delta q$  in the reference dynamics (32) continue to ensure bounded tracking errors.

To implement the proposed controller, we follow these steps:



FIGURE 1. Ur5e industrial robot.

- 1) Define the desired trajectory.
- 2) Initialize the neural network weights  $\hat{G}_d(0)$ ,  $\hat{G}_U(0)$ , and  $\hat{G}_\lambda(0)$  to zero.
- 3) Set the control parameters  $\Gamma_d$ ,  $\Gamma_U$ ,  $\Gamma_\lambda$ ,  $k_g$ ,  $\hat{G}(0)$ ,  $\kappa_1$ ,  $\kappa_0$ , and  $l_j$ .
- 4) Compute  $\dot{q}_r$  using (32).
- 5) Calculate the activation functions  $\Theta_d$  and  $\Theta_U$ .
- 6) Compute  $\lambda$ ,  $\dot{\lambda}$ , and  $\int_0^t \lambda(\tau) d\tau$  using (8). Then, calculate  $\Theta_\lambda$ .
- 7) Determine the vector  $q_r^* = \dot{q}_r - \hat{G}\hat{G}_\lambda\Theta_\lambda$ .
- 8) Calculate the sliding vector  $s$  using (11).
- 9) Apply the joint velocity (or position) command signal according to (18).
- 10) Update the weight matrices  $\hat{G}_d$ ,  $\hat{G}_U$ , and  $\hat{G}_\lambda$  using the update laws in (19), (20), and (21), respectively. Also, update the estimate of  $\hat{G}$  in (22).
- 11) Repeat from step 4 to 10 until the desired task is achieved.

*Theorem 2:* If the gain parameter  $\gamma$  is chosen as  $\gamma = \hat{G}$ , and  $\hat{G}_\lambda\Theta_\lambda$  represents an approximation of the term  $\lambda + \int_0^t \lambda(\tau) d\tau$ . Additionally, the control parameter  $k_g$  is chosen such that  $k_g > b$ . Then, by employing the joint velocity (or position) command signal described in (18), along with adaptive neural network-based update laws in (19)-(21) and the adaptive law in (22), in conjunction with the reference dynamics provided in (32), we ensure the convergence of the joint tracking errors, i.e.,  $\Delta q \rightarrow 0$  and  $\Delta \dot{q} \rightarrow 0$  as  $t \rightarrow \infty$ .

*Proof:* According to (30),  $\hat{G}_\lambda$  is bounded, and by definition in (2), the activation function  $\Theta_\lambda$  is also bounded. With (32), the subsystem (25) of the differential cascaded system changes to

$$\Delta \ddot{q} = -\kappa_1 \Delta \dot{q} - \kappa_0 \Delta q + \ddot{\lambda} + \hat{G}_\lambda \Theta_\lambda \quad (33)$$

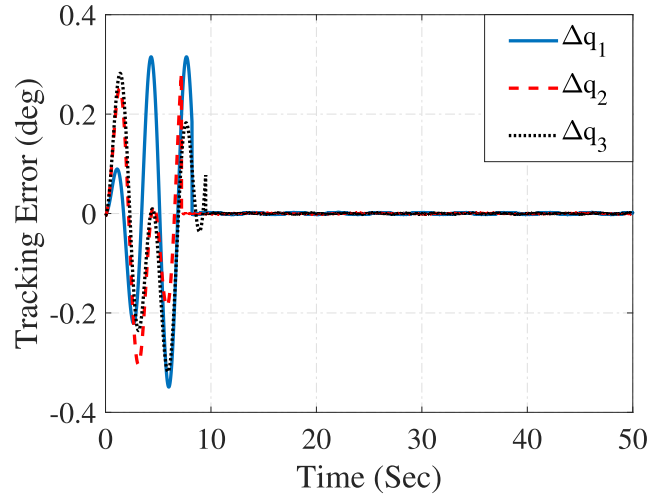


FIGURE 2. Position tracking errors of the proposed neural network-based controller.

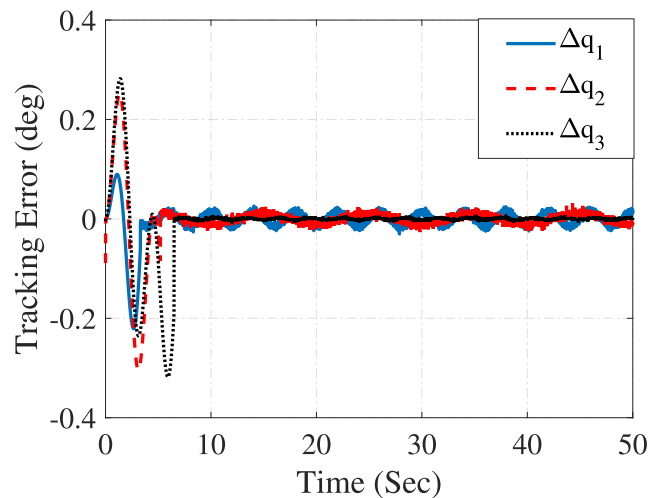


FIGURE 3. Position tracking errors of the controller in [8].

with output as  $x = [\Delta q^T, \Delta \dot{q}^T]^T$  and inputs as  $y_1 = \ddot{\lambda}$  and  $y_2 = \hat{G}_\lambda\Theta_\lambda$ . According to Theorem 1,  $\int_0^t y_1(\tau) d\tau = \int_0^t \ddot{\lambda}(\tau) d\tau = \dot{\lambda} - \dot{\lambda}(0) \in \mathcal{L}_2 \cap \mathcal{L}_\infty$ , and we also establish that  $y_2 \in \mathcal{L}_\infty$ . Employing analogous reasoning to that of Theorem 1, we deduce that  $x \in \mathcal{L}_\infty$ . Consequently,  $\dot{q}_r \in \mathcal{L}_\infty$ . Demonstrating the boundedness of other control signals follows a procedure akin to Theorem 1.  $\square$

#### IV. EXPERIMENTAL RESULTS

The proposed control system was implemented on the UR5e industrial manipulator (see Fig. 1) for experimental validation. The manipulator's inner control loop, which governs its operation, remains fixed and undisclosed to the user. Consequently, users can only specify inputs to the robot through joint position or velocity commands. To streamline the robot's movement, we directed our attention toward its primary axes, which include the base, shoulder, and elbow, while leaving the minor axes unchanged. The desired joint

position is defined as  $q_d = [(\frac{\pi}{18})[1 - \cos(\frac{2\pi t}{3})], \frac{\pi}{2} + \frac{\pi}{18}[1 - \cos(\frac{2\pi t}{3})], \frac{\pi}{18}[1 - \cos(\frac{2\pi t}{3})]]^T$  (in radians). A sampling frequency of 125 Hz was employed during the experiment.

We chose 20 neurons for each of the neural network so that  $k_g$  can be assigned as zero. The parameters of the update laws in (19-22) are specified as follows:  $\Gamma_d^{-1} = 0.3 * I_3$ ,  $\hat{G} = 0.3 * I_3$ ,  $\Gamma_U^{-1} = 5 * I_3$ ,  $\Gamma_\lambda^{-1} = 8 * I_3$  and  $l_j = 10$ ,  $j = 1, 2, 3$ . The weights of neural network were initialized as  $\hat{G}_d = 0_{3 \times 20}$ ,  $\hat{G}_U = 0_{3 \times 20}$  and  $\hat{G}_\lambda = 0_{3 \times 20}$ . The positive design constants  $\kappa_1, \kappa_0$  are selected as  $\kappa_1 = 2$  and  $\kappa_0 = 3$ .

The effectiveness of the designed user-defined velocity (or position) command signals in enabling the closed architecture robot manipulator to track the desired joint positions is demonstrated in Figure 2. Initially, the transient response appears relatively low due to the neural network weights being initialized to zero, compared to the position tracking error obtained by the adaptive-based controller in [8], as shown in Figure 3. However, as the neural network weights are updated, the response of the designed controller becomes more efficient, surpassing its counterpart in [8], which exhibits fluctuations in its steady-state response.

## V. CONCLUSION

The adaptive neural network framework proposed in this study demonstrates promising results in addressing the complexities associated with controlling closed architecture industrial robot manipulators. The approach, which operates independently of inner controller configuration and control gain structure, exhibits comprehensive control variable boundedness and global applicability across varying conditions. Moving forward, future research endeavors will focus on extending this framework to delve into task-space control with a particular emphasis on achieving prescribed performance bounds. This expansion aims to further enhance the adaptability and efficiency of the proposed control strategy in practical industrial settings.

## REFERENCES

- [1] F. Lange and G. Hirzinger, "Learning to improve the path accuracy of position controlled robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 1994, pp. 494–501.
- [2] M. Grotjahn and B. Heimann, "Model-based feedforward control in industrial robotics," *Int. J. Robot. Res.*, vol. 21, no. 1, pp. 45–60, Jan. 2002.
- [3] H. Wang, W. Ren, C. C. Cheah, Y. Xie, and S. Lyu, "Dynamic modularity approach to adaptive control of robotic systems with closed architecture," *IEEE Trans. Autom. Control*, vol. 65, no. 6, pp. 2760–2767, Jun. 2020.
- [4] S. Lyu and C. C. Cheah, "Human-robot interaction control based on a general energy shaping method," *IEEE Trans. Control Syst. Technol.*, vol. 28, no. 6, pp. 2445–2460, Nov. 2020.
- [5] S. Lyu and C. C. Cheah, "Data-driven learning for robot control with unknown Jacobian," *Automatica*, vol. 120, Oct. 2020, Art. no. 109120.
- [6] G. D. Khan, H.-T. Nguyen, and C. C. Cheah, "A stable control strategy for industrial robots with external feedback loop," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 12833–12838.
- [7] H. Wang, Y. Li, and T. Jiang, "Bilateral control of teleoperators with closed architecture and time-varying delay," 2021, *arXiv:2106.12470*.
- [8] H. Wang and Y. Li, "Adaptive control of robot manipulators with closed architecture," *Automatica*, vol. 162, Apr. 2024, Art. no. 111040.
- [9] G. D. Khan, "Control of robot manipulators with uncertain closed architecture using neural networks," *Intell. Service Robot.*, vol. 17, no. 2, pp. 315–327, Mar. 2024.

- [10] S. Arimoto, *Control Theory of Non-Linear Mechanical Systems: A Passivity-Based and Circuit-Theoretic Approach*. London, U.K.: Oxford Univ. Press, 1996.
- [11] A. Astolfi and R. Ortega, "Immersion and invariance: A new tool for stabilization and adaptive control of nonlinear systems," *IEEE Trans. Autom. Control*, vol. 48, no. 4, pp. 590–606, Sep. 2003.
- [12] A. Loría, "Observers are unnecessary for output-feedback control of Lagrangian systems," *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 905–920, Apr. 2016.
- [13] J. J. Craig, *Introduction to Robotics: Mechanics and Control, 3/E*. London, U.K.: Pearson, 2009.
- [14] J. J. Craig, P. Hsu, and S. S. Sastry, "Adaptive control of mechanical manipulators," *Int. J. Robot. Res.*, vol. 6, no. 2, pp. 16–28, 1987.
- [15] J. J. Slotine and W. Li, "On the adaptive control of robot manipulators," *Int. J. Robot. Res.*, vol. 6, no. 3, pp. 49–59, Sep. 1987.
- [16] F. L. Lewis, K. Liu, and A. Yesildirek, "Neural net robot controller with guaranteed tracking performance," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 703–715, May 1995.
- [17] R. M. Sanner and J. E. Slotine, "Stable adaptive control of robot manipulators using 'neural' networks," *Neural Comput.*, vol. 7, no. 4, pp. 753–790, Jul. 1995.
- [18] X. Li and C. C. Cheah, "Adaptive neural network control of robot based on a unified objective bound," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 3, pp. 1032–1043, May 2014.
- [19] M. W. Spong, "On the robust control of robot manipulators," *IEEE Trans. Autom. Control*, vol. 37, no. 11, pp. 1782–1786, Nov. 1992.
- [20] H. Wang, "Differential-cascade framework for consensus of networked Lagrangian systems," *Automatica*, vol. 112, Feb. 2020, Art. no. 108620.
- [21] C. A. Desoer and M. Vidyasagar, *Feedback Systems: Input-Output Properties*. Philadelphia, PA, USA: SIAM, 2009.



**GULAM DASTAGIR KHAN** was born in India. He received the degree in engineering, the bachelor's degree in electronics and instrumentation engineering from Osmania University, India, in 2009, the master's degree in control instrumentation and systems engineering from the King Fahd University of Petroleum and Minerals, Saudi Arabia, in 2013, and the Ph.D. degree in electrical engineering from The University of Newcastle, Australia, in 2019. He was a Lecturer with universities in India and Saudi Arabia. Additionally, he gained research experience as a Postdoctoral Researcher with the Robotics Laboratory, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Currently, he is an Assistant Professor with the Department of Electrical Engineering, Sultan Qaboos University, Oman. He has a background in teaching and research. His research interests include networked control systems, event-driven systems, closed-architecture robotic systems, and micro-manipulation.

• • •