Adaptive neural network control of robot based on a unified objective bound

Li, Xiang; Cheah, Chien Chern

2013

Li, X., & Cheah, C. C. (2013). Adaptive neural network control of robot based on a unified objective bound. IEEE transactions on control systems technology, PP(99), 1-12.

<https://hdl.handle.net/10356/101658>

© 2013 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at:
[<http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6684277&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel>]

Downloaded on 26 Apr 2023 22:42:35 SGT

Adaptive Neural Network Control of Robot based on a Unified Objective Bound

Xiang Li, *Member, IEEE*, and Chien Chern Cheah, *Senior Member, IEEE*,

Abstract—In the conventional adaptive neural network control of robotic manipulator, the desired position of robot end effector is specified as a point or trajectory. In addition, it is usually difficult to guarantee the transient performance of adaptive neural network control system due to the initialization error of the weight of neural network. In this paper, a new control formulation is proposed for the adaptive neural network control of robotic manipulator, which unifies existing neural network control tasks such as setpoint control, trajectory tracking control and trajectory tracking control with prescribed performance bound. The proposed method also includes a new adaptive neural network control scheme where the objective for the robot end effector can be specified as a dynamic region, instead of the desired position or trajectory. The stability of the closed-loop system is analyzed by using Lyapunov-like analysis. Experimental results are presented to illustrate the performance of the proposed approach and the energy-saving property of the proposed neural network controller with dynamic region.

Index Terms—Adaptive neural network, robot control, unified bound, performance bound.

I. INTRODUCTION

ROBOT dynamics is highly nonlinear with couplings existing between joints. Several model-free controllers [1]–[3] have been developed for setpoint control of robotic manipulator despite the nonlinearity of the dynamics. A path tracking controller was also developed in [2], which guarantees the uniform ultimate boundedness of velocity error. To deal with the trajectory tracking control problem with uncertain dynamic parameters, much effort has been devoted to developing adaptive control schemes for robotic manipulator and much progress has been achieved in understanding how the robot can deal with the dynamic uncertainty [3]–[9]. In adaptive robot control, the convergence of task error can be guaranteed without the convergence of estimated parameters, and the desired position of robot is usually specified as a setpoint or trajectory. Instead of limiting the desired task to a position or trajectory, the concept of task-space region control [10]–[12] where the desired objective is specified as a region was also proposed.

One common assumption in the robot adaptive control is that the structure of the robot dynamics is known, and the construction of a regressor matrix is necessary. Neural network has a well-known property that it can approximate arbitrary non-linear functions and learn through examples, and hence it allows robot control without structure assumed in the

mentioned adaptive control laws. Scanner and Slotine [13]–[15] developed several control strategies using various neural networks such as Gaussian networks and dynamically structured networks for nonlinear system. Lewis *et al.* [16]–[18] proposed several adaptive neural network controllers for robotic manipulator, where the weight of neural networks was updated without preliminary offline training. Some systematic approaches for structured dynamic modeling and adaptive robot controllers using neural network can also be found in [19]. A feedforward adaptive controller was proposed for robotic manipulator in [20], where the feedforward signal was constituted by outputs of a set of fixed multilayer neural networks. A neuro controller was developed for robotic manipulator in [21], which consisted of a set of off-line trained neural networks to deal with uncertain robot dynamics and payload. An adaptive output feedback control method was proposed for flexible-joint robots in [22], where an observer using neural networks was introduced to eliminate the requirement of velocity measurement. In [23], the neural-network technique was employed to compensate the uncertainties in the manipulator dynamics and actuator model. In [24], an adaptive neural network Jacobian controller was developed for multifingered robot hands with uncertain kinematics, Jacobian matrices and dynamics. An adaptive neural network controller was developed for robot interacting with a viscoelastic environment in [25]. An adaptive neural network control approach was introduced for contouring control of manipulator in [26].

The main advantage of the adaptive neural network control is that the stability and convergence can be ensured without preliminary offline training. The weight of neural network is adjusted online by an update law but it is usually hard to initialize the weight. The initialization error affects the transient response of the adaptive neural network control system, which may degrade the performance of system or even compromise the safety of users. To solve the problem, an adaptive controller with prescribed performance [27] was recently proposed to guarantee the transient response of robot systems.

Existing adaptive neural network control methods for robotic manipulator are focusing on setpoint or tracking control, where the desired position of robot end effector is always specified as a point or trajectory, and the position errors are used to drive the end effector towards the desired position. In this paper, a new adaptive neural network control formulation is proposed for robotic manipulator, where the objective for the robot end effector can be specified as a unified bound. The unified bound is a general objective function for various robotic manipulation tasks. When the bound is specified arbitrarily

small, it reduces to the conventional setpoint or trajectory. When the bound is specified as the performance bound, the transient and steady-state response of closed-loop system can be guaranteed. It can also be specified as a general dynamic region that can be scaled or rotated to provide flexibility for the specification of robot tasks. Thus the proposed method also includes a new adaptive neural network control scheme with dynamic region. The convergence to the unified bound is proved by using Lyapunov-like analysis. Experimental results are presented to illustrate the performance of the proposed approach and the energy-saving property of the proposed neural network controller with the dynamic region.

II. BACKGROUND

A. Robot Kinematics and Dynamics

Let $\mathbf{x} = [x_1, \dots, x_m]^T \in \mathbb{R}^m$ denotes the position of the end effector in task space, and $\mathbf{q} = [q_1, \dots, q_n]^T \in \mathbb{R}^n$ is a vector of joint angles. The velocity of the end effector in task space $\dot{\mathbf{x}}$ is related to the joint-space velocity $\dot{\mathbf{q}}$ as:

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \quad (1)$$

where $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{m \times n}$ is the Jacobian matrix from joint space to task space. The dynamics of the robotic manipulator is described as:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + [\frac{1}{2}\dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})]\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}, \quad (2)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n \times n}$ is an inertia matrix which is symmetric and positive definite, $\mathbf{S}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n \times n}$ is a skew-symmetric matrix, $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^n$ denotes a vector of gravitational force, and $\boldsymbol{\tau} \in \mathbb{R}^n$ denotes a vector of control input.

B. Neural Networks

An important property of neural networks is the ability to approximate an arbitrary nonlinear function up to a small error. In order to represent the function, an approximating function is chosen first, then the weights are updated based on the output errors. In this paper, the radial basis function (RBF) network is used to approximate the uncertainties in the dynamic model of robot and the weights of neural network are adjusted online without previous learning phase.

The structure of a RBF neural network is shown in Fig. 1. In the RBF neural network, $\mathbf{p} = [p_1, \dots, p_{n_p}]^T \in \mathbb{R}^{n_p}$ is the input of the network, and $\mathbf{h}(\mathbf{p})$ is the output, and $\boldsymbol{\theta} = [\theta_1, \dots, \theta_{n_a}]^T \in \mathbb{R}^{n_a}$ is the activation function, where $\theta_i (i = 1, \dots, n_a)$ are specified as:

$$\theta_i = e^{-\frac{\|\mathbf{p} - \boldsymbol{\mu}_i\|^2}{\sigma_i^2}}, \quad (3)$$

where $\boldsymbol{\mu}_i = [\mu_{i1}, \dots, \mu_{in_p}]^T \in \mathbb{R}^{n_p}$ is called the center, and $\boldsymbol{\sigma} = [\sigma_1, \dots, \sigma_{n_a}]^T \in \mathbb{R}^{n_a}$ is called the distance and $\sigma_i > 0$. From Fig. 1, the function approximation using a RBF neural network is:

$$\mathbf{h}(\mathbf{p}) = \mathbf{W}\boldsymbol{\theta}(\mathbf{p}) + \mathbf{E}, \quad (4)$$

where $\mathbf{W}$ is the weight matrix, and $\mathbf{E}$ is called neural network functional approximation error, and the error generally decreases when the number of neurons increases.

Fig. 1. A RBF neural network.

III. UNIFIED BOUND

In this paper, the objective for the robot end effector is specified as a unified bound instead of a setpoint or trajectory, to suit different applications of robotic manipulation.

A. Dynamic Region

When the unified bound is specified as a dynamic region, the desired region can be scaled and rotated to allow flexibility for robot movement. The dynamic region is described by the following inequality functions:

$$\mathbf{f}(\Delta\mathbf{x}_T) = [f_1(\Delta\mathbf{x}_T), \dots, f_M(\Delta\mathbf{x}_T)]^T \leq \mathbf{0}, \quad (5)$$

where M is the total number of objective functions. The variable $\Delta\mathbf{x}_T$ in equation (5) is defined as:

$$\Delta\mathbf{x}_T = \bar{\mathbf{T}}(\mathbf{x} - \mathbf{x}_f) = \bar{\mathbf{T}}\Delta\mathbf{x}, \quad (6)$$

where $\bar{\mathbf{T}}$ is a transformation matrix, $\mathbf{x}_f = [x_{f1}, \dots, x_{fm}]^T \in \mathbb{R}^m$ is a reference position, and $\Delta\mathbf{x} = \mathbf{x} - \mathbf{x}_f$ represents the position error with respect to the reference position. As seen from equation (6), $\Delta\mathbf{x}_T$ is a product of the transformation matrix $\bar{\mathbf{T}}$ and the position error $\Delta\mathbf{x}$, which provides directional information towards the time-varying dynamic region.

The vector inequality $\mathbf{f}(\Delta\mathbf{x}_T) \leq \mathbf{0}$ implies that $f_i(\Delta\mathbf{x}_T) \leq 0$ for all $i = 1, \dots, M$. The functions $f_i(\Delta\mathbf{x}_T)$ are scalar functions with continuous partial derivatives, and $f_i(\Delta\mathbf{x}_T)$ specifies a moving region where the reference position $\mathbf{x}_f$ is time-varying.

The transformation matrix $\bar{\mathbf{T}}$ should be specified such that the inverse matrix $\bar{\mathbf{T}}^{-1}$ exists. In addition, it can be specified as a scaling matrix or a rotation matrix according to specific robot tasks. For example, $\bar{\mathbf{T}}$ can be specified as a scaling matrix in 3-D space as:

$$\bar{\mathbf{T}} = \mathbf{S} = \begin{bmatrix} \frac{1}{S_1} & 0 & 0 \\ 0 & \frac{1}{S_2} & 0 \\ 0 & 0 & \frac{1}{S_3} \end{bmatrix}, \quad (7)$$

where $S_i(t) (i = 1, 2, 3)$ are positive scaling factors, and $\mathbf{S}$ is a symmetric nonsingular matrix. Therefore, we have:

$$\Delta\mathbf{x}_T = \bar{\mathbf{T}} \begin{bmatrix} x_1 - x_{f1} \\ x_2 - x_{f2} \\ x_3 - x_{f3} \end{bmatrix} = \begin{bmatrix} \frac{x_1 - x_{f1}}{S_1} \\ \frac{x_2 - x_{f2}}{S_2} \\ \frac{x_3 - x_{f3}}{S_3} \end{bmatrix}. \quad (8)$$

Various dynamic regions such as sphere, cube, cylinder etc. can be formed by choosing the appropriate functions and using the scaling matrix in equation (7). For example, when the desired region is specified as a cube in 3-D space, the inequality functions in equation (5) can be specified as:

$$\begin{aligned} f_1(\Delta \mathbf{x}_T) &= \frac{(x_1 - x_{f1})^2}{(S_1 b_1)^2} - 1 \leq 0, \\ f_2(\Delta \mathbf{x}_T) &= \frac{(x_2 - x_{f2})^2}{(S_2 b_2)^2} - 1 \leq 0, \\ f_3(\Delta \mathbf{x}_T) &= \frac{(x_3 - x_{f3})^2}{(S_3 b_3)^2} - 1 \leq 0, \end{aligned} \quad (9)$$

where $b_i (i = 1, 2, 3)$ are the individual regional bounds for each axis. When the desired region is designed as a sphere, the inequality function in equation (5) can be specified as:

$$f(\Delta \mathbf{x}_T) = \frac{(x_1 - x_{f1})^2}{(S_1 r)^2} + \frac{(x_2 - x_{f2})^2}{(S_2 r)^2} + \frac{(x_3 - x_{f3})^2}{(S_3 r)^2} - 1 \leq 0, \quad (10)$$

where r is the radius of the sphere, and $S_1 = S_2 = S_3 = S$. The size of the region increases as the factors $S_i (i = 1, 2, 3)$ increase, and vice versa.

The transformation matrix $\bar{T}$ can also be specified as a rotation matrix or a composite rotation matrix. For example:

$$\bar{T} = \mathbf{R} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix}, \quad (11)$$

where $\phi(t)$ and $\varphi(t)$ are rotation angles.

In addition, the transformation matrix $\bar{T}$ can be specified as the product of the scaling matrix and the rotation matrix: $\bar{T} = \mathbf{R}\mathbf{S}$, so as to allow the desired region to be scaled and rotated simultaneously. Some examples of dynamic regions are illustrated in Fig. 2.

Fig. 2. Dynamic regions before and after scaling and rotation.

Therefore, the size of the dynamic region can be varied by specifying the transformation matrix. In an application where the high precision is required, it is possible to define the region arbitrarily small. When the precision is not critical, the region could be scaled up to provide more flexibility in task specifications. Fig. 3 illustrates a scenario of manipulation by using the dynamic region.

B. Performance Bound

It is known that the initialization error of weight of neural network may affect the transient performance of robot system. To solve the problem, the concept of prescribed performance has been proposed in [27]–[30] to guarantee the transient response of closed-loop system. In this paper, the unified bound can also be formulated as a performance bound, such that the transient and steady-state response of closed-loop system is guaranteed when the end effector stays inside the performance bound.

When the unified bound is specified as the performance bound, the function $f_i(\Delta \mathbf{x}_T)$ denotes the bound in the i^{th} coordinate. As the performance bound shrinks to a bound around the reference position $\mathbf{x}_f$, the tracking error $\Delta \mathbf{x} = \mathbf{x} - \mathbf{x}_f$ reduces correspondingly. Therefore, the variation of the tracking error is related with the variation of the performance bound, and the transformation matrix $\bar{T}$ and the specific form of the function $f_i(\Delta \mathbf{x}_T)$ determine the different transient performance of closed-loop system.

For example, when a transient response with an allowable maximum overshoot as illustrated in Fig. 4.(a) is required, the region function $f_i(\Delta \mathbf{x}_T)$ can be specified as:

$$f_i(\Delta \mathbf{x}_T) = \frac{\Delta x_{Ti}^2}{b_i^2} - 1 \leq 0, \quad (12)$$

where

$$b_i = \begin{cases} b_{ci}, & x_i \geq x_{fi}, \\ -M_p b_{ci}, & x_i < x_{fi}, \end{cases} \quad (13)$$

where $0 < M_p < 1$ is the allowable overshoot, b_i denote the positions of boundaries, and b_{ci} are constants. The transformation matrix $\bar{T}$ is specified as a $m \times m$ matrix as:

$$\bar{T} = \begin{bmatrix} \frac{1}{\frac{e_{ss}}{M_p b_{c1}} + e^{-l_c t}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\frac{e_{ss}}{M_p b_{cm}} + e^{-l_c t}} \end{bmatrix}, \quad (14)$$

where l_c represents the convergence speed of bounds, and e_{ss} represents the steady-state error. From equations (13) and (14), the initial upper bounds are obtained as: $b_{max} = b_{ci} + \frac{e_{ss}}{M_p}$, and the initial lower bounds are obtained as: $b_{min} = -M_p b_{ci} - e_{ss}$.

The initial position of the end effector is either measured with the sensory feedback, or computed from the forward kinematic equations by using the initial joint configurations measured from encoders. After the initial position is determined, the initial upper and lower bounds can be set to enclose the initial position with respect to the reference position, such that the end effector is initially inside the performance bound as illustrated in Fig. 4.

When a transient response without overshoot as illustrated in Fig. 4.(b) is required, $f_i(\Delta \mathbf{x}_T)$ is specified as:

$$f_i(\Delta \mathbf{x}_T) = \left[\frac{\Delta x_{Ti}^2}{b_{ui}^2} - 1 \right] \leq 0, \quad (15)$$

where b_{ui} and b_{li} are constants. The transformation matrix $\bar{T}$

Fig. 3. A scenario of robotic manipulation by using a scaling dynamic region: (a) The region is set small such that the end effector can exactly grasp the desired object on a table; (b) After the object has been grasped, the dynamic region is then scaled up such that the end effector can place the object at any position on another table.

is specified as:

$$\bar{T} = \begin{bmatrix} \frac{1}{\frac{e_{ss}}{b_{li}} + e^{-l_c t}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\frac{e_{ss}}{b_{lm}} + e^{-l_c t}} \end{bmatrix}. \quad (16)$$

From equations (15) and (16), the initial upper bounds are obtained as: $b_{\max} = b_{ui} + e_{ss} \frac{b_{ui}}{b_{li}}$, and the initial lower bounds are obtained as: $b_{\min} = b_{li} + e_{ss}$. Similarly, the upper and lower bounds are set such that the end effector is inside the performance bound from the beginning and stays within it. The overshoot is eliminated when the end effector is restricted where $f_i(\Delta x_T) \leq 0$. As the performance bound shrinks, the end effector is driven towards the reference position.

Fig. 4. An illustration of transient response: (a) with allowable maximum overshoot; (b) without overshoot.

As seen from section III.A-III.B, the proposed unified bound is a generalization of setpoint, trajectory, performance bound, and dynamic region. The specifications of the bound according to various objectives are summarized as follows:

(i) When the bound is specified arbitrarily small and the

reference position is the desired constant point ($\dot{x}_f = 0$), it reduces to a setpoint;

(ii) When the bound is specified arbitrarily small and the reference position is time-varying ($\dot{x}_f \neq 0$), it reduces to a trajectory;

(iii) When the dynamic region is specified to enclose the initial error, the desired transient performance can be guaranteed.

Remark 1: The prescribed performance is defined specifically to guarantee the transient and steady-state response of a control system, while the dynamic region is more general and flexible. It can be rotated, scaled up and down to ensure accuracy and allow flexibility of the task. It can also be specified as the conventional setpoint or trajectory to suit different robotic manipulation tasks. If the dynamic region is specified to enclose the initial error and the end effector is controlled to stay inside the region throughout the robot movement, the prescribed performance of the control system is also guaranteed. $\triangle\triangle\triangle$

C. Potential Energy

The potential energy function for the unified bound is proposed as:

$$P(\Delta x_T) = \sum_{i=1}^M P_i(\Delta x_T), \quad (17)$$

where

$$P_i(\Delta x_T) = \frac{k_{xi}}{2} [\max(0, f_i(\Delta x_T))]^2. \quad (18)$$

That is,

$$P_i(\Delta x_T) = \begin{cases} 0, & f_i(\Delta x_T) \leq 0, \\ \frac{k_{xi}}{2} [f_i(\Delta x_T)]^2, & f_i(\Delta x_T) > 0, \end{cases} \quad (19)$$

where k_{xi} are positive constants. The above energy function is smooth and lower bounded by zero.

Partial differentiating the potential energy function described by (18) with respect to Δx_T yields:

$$\left(\frac{\partial P_i(\Delta x_T)}{\partial \Delta x_T} \right)^T = \begin{cases} 0, & f_i(\Delta x_T) \leq 0, \\ k_{xi} f_i(\Delta x_T) \left(\frac{\partial f_i(\Delta x_T)}{\partial \Delta x_T} \right)^T, & f_i(\Delta x_T) > 0. \end{cases} \quad (20)$$

From equation (17), we have

$$\begin{aligned} \left(\frac{\partial P(\Delta x_T)}{\partial \Delta x_T} \right)^T &= \sum_{i=1}^M \left(\frac{\partial P_i(\Delta x_T)}{\partial \Delta x_T} \right)^T \\ &= \sum_{i=1}^M k_{xi} \max(0, f_i(\Delta x_T)) \left(\frac{\partial f_i(\Delta x_T)}{\partial \Delta x_T} \right)^T, \end{aligned} \quad (21)$$

which denotes the gradient of the potential energy. When the end effector enters the bound, $f_i(\Delta \mathbf{x}_T) < 0$, and the gradient automatically reduces to zero, as illustrated in Fig. 5.

Fig. 5. The gradient is nonzero when the end effector is outside the bound, and it automatically reduces to zero after the end effector enters the bound.

It is intuitive that reaching for bounds instead of setpoint or trajectory would allow more flexibility in robot tasks. In fact, the potential energy $P_i(\Delta \mathbf{x}_T)$ remains zero if the end effector starts from an initial position within the bound and stays inside the bound where $f_i(\Delta \mathbf{x}_T) < 0$. Comparatively, the potential energy using the conventional position error $\Delta \mathbf{x}$ is nonzero until the end effector reaches the reference position $\mathbf{x}_f$. Fig. 6 illustrates the variation of potential energy $P_i(\Delta \mathbf{x}_T)$ as the desired bound shrinks. In Fig. 6, $P_i(\Delta \mathbf{x}_T)$ is zero when the end effector is inside the desired bound, and the bottom of $P_i(\Delta \mathbf{x}_T)$ shrinks to a point as the bound is scaled down.

When the unified bound is specified as a performance bound, the potential energy should be specified with an arbitrarily large gradient to keep the position error within the performance bound. To fulfill the requirement, a high-order potential energy $P_h(\Delta \mathbf{x}_T)$ is introduced as:

$$P_h(\Delta \mathbf{x}_T) = \sum_{i=1}^M \frac{k_{hi}}{N} [\max(0, f_i(\Delta \mathbf{x}_T))]^N, \quad (22)$$

where k_{hi} are positive constants, and $N \geq 10$ is the order of the function which is also an even integer. The potential energy is then the combination of $P_h(\Delta \mathbf{x}_T)$ and the original potential energy $P(\Delta \mathbf{x}_T)$ as:

$$P_c(\Delta \mathbf{x}_T) = P_h(\Delta \mathbf{x}_T) + P(\Delta \mathbf{x}_T), \quad (23)$$

where $P_c(\Delta \mathbf{x}_T)$ represents the new potential energy. An illustration of $P_c(\Delta \mathbf{x}_T)$ is shown in Fig. 7(a). Note that the parameters k_{xi} and k_{hi} in equation (23) are set large to create the steep gradient.

The steep gradient of $P_c(\Delta \mathbf{x}_T)$ may cause the oscillation of robot movement when the end effector is very near the boundary of the performance bound. To alleviate the problem, another potential energy function with a smaller reference bound can be introduced as:

$$P_r(\Delta \mathbf{x}_T) = \sum_{i=1}^M \frac{k_{ri}}{2} [\max(0, f_{ri}(\Delta \mathbf{x}_T))]^2, \quad (24)$$

where k_{ri} are positive constants, and $f_{ri}(\Delta \mathbf{x}_T) \leq 0$ is the reference bound which is inside $f_i(\Delta \mathbf{x}_T) \leq 0$. The parameters k_{ri} are not large and thus the gradient of $P_r(\Delta \mathbf{x}_T)$ is smaller.

Therefore, the overall potential energy can be proposed as the summation of $P_r(\Delta \mathbf{x}_T)$ and $P_c(\Delta \mathbf{x}_T)$ as:

$$P_o(\Delta \mathbf{x}_T) = P_c(\Delta \mathbf{x}_T) + P_r(\Delta \mathbf{x}_T), \quad (25)$$

where $P_o(\Delta \mathbf{x}_T)$ represents the overall potential energy. An illustration of $P_o(\Delta \mathbf{x}_T)$ is shown in Fig. 7(b). If the end effector moves beyond the reference region where $f_{ri}(\Delta \mathbf{x}_T) > 0$, the gradient of $P_r(\Delta \mathbf{x}_T)$ becomes nonzero and drives the end effector back from the steep gradient. The reference region is hence introduced to reduce the possibility of oscillation in actual implementations. Note that the use of reference region is optional.

(a) $P_c(\Delta \mathbf{x}_T)$ (without $f_{ri}(\Delta \mathbf{x}_T)$) (b) $P_o(\Delta \mathbf{x}_T)$ (with $f_{ri}(\Delta \mathbf{x}_T)$)

Fig. 7. Examples of potential energy functions with or without the reference region. Both $P_c(\Delta \mathbf{x}_T)$ and $P_o(\Delta \mathbf{x}_T)$ are smooth and lower-bounded by zero, and the gradient of the potential energy $P_r(\Delta \mathbf{x}_T)$ for the reference region becomes nonzero when the end effector moves beyond the reference region, so as to drive it back from the steep gradient.

Since the concept of the neural network control with the unified bound is general, it is also possible to formulate other potential energy functions with steep gradient, to keep the tracking errors inside the performance bound. For example, another potential energy function for the performance bound can be specified as:

$$P_o(\Delta \mathbf{x}_T) = \sum_{i=1}^M \frac{k_{xi}}{2} \frac{[\max(0, f_{ri}(\Delta \mathbf{x}_T))]^2}{[f_i(\Delta \mathbf{x}_T)]^2}, \quad (26)$$

where an illustration of the potential energy $P_o(\Delta \mathbf{x}_T)$ in equation (26) is shown in Fig. 8. Since the initial bound encloses the initial error such that $f_{ri}(\Delta \mathbf{x}_T) < 0$, the initial potential energy is zero. When the end effector is very near the boundary where $f_i(\Delta \mathbf{x}_T) = 0$, the gradient of the potential energy becomes very large.

Similarly, partial differentiating $P_o(\Delta \mathbf{x}_T)$ with respect to $\Delta \mathbf{x}_T$ yields:

$$\left(\frac{\partial P_o(\Delta \mathbf{x}_T)}{\partial \Delta \mathbf{x}_T} \right)^T \triangleq \Delta \boldsymbol{\varepsilon}, \quad (27)$$

where $\Delta \boldsymbol{\varepsilon}$ denotes a unified region error which is also the gradient of $P_o(\Delta \mathbf{x}_T)$. When the unified bound is not specified as the performance bound, k_{hi} and k_{ri} are set as zero, and $\Delta \boldsymbol{\varepsilon}$ smoothly transits from nonzero to zero after the end effector enters the bound. When the unified bound is specified as the performance bound, k_{xi} and k_{hi} are set large while k_{ri} can be set small, and $\Delta \boldsymbol{\varepsilon}$ becomes arbitrarily large if the end effector is outside the performance bound. The parameters of the region error are selected according to the specific bound as shown in Table I.

Fig. 6. The bottom of potential energy corresponds to the bound. As the bound shrinks, the bottom of potential energy converges to the desired position.

TABLE I
PARAMETER SELECTION OF UNIFIED REGION ERROR

TYPE	SIZE OF BOUND	INITIAL BOUND	$\mathbf{x}_f$	$\mathbf{T}$	k_{xi}	k_{hi}
Setpoint	arbitrarily small	not required	desired point	identity matrix	nonzero	zero
Trajectory	arbitrarily small	not required	desired trajectory	identity matrix	nonzero	zero
Static region	nonzero and constant	not required	constant position inside region	constant	nonzero	zero
Dynamic region	time-varying	not required	time-varying position inside region	scaling or rotation matrix	nonzero	zero
Performance bound	from initial bound to small bound	enclosing initial position error	constant or time-varying	based on transient requirement	arbitrarily large	large

Fig. 8. An illustration of $P_o(\Delta \mathbf{x}_T)$. The potential energy is smooth and lower-bounded by zero, and the gradient of the potential energy becomes steep at the boundary of the performance bound.

IV. ADAPTIVE NEURAL NETWORK CONTROL

In this section, we propose a new adaptive neural network controller. The desired objective is specified as a unified bound to provide flexibility in robot tasks. When the bound is specified arbitrarily small, the control objective reduces to the conventional trajectory tracking control. When the unified bound is specified as a performance bound, the transient response of the robot system is guaranteed.

The proposed controller is specified as:

$$\begin{aligned} \boldsymbol{\tau} = & -\mathbf{K}_s \mathbf{s} - \mathbf{J}^T(\mathbf{q}) \bar{\mathbf{T}}^T \Delta \boldsymbol{\varepsilon} \\ & - k_g \text{sgn}(\mathbf{s}) + \hat{\mathbf{W}}_d \boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f), \end{aligned} \quad (28)$$

where $\mathbf{s}$ is a sliding vector which is introduced as:

$$\begin{aligned} \mathbf{s} = & \dot{\mathbf{q}} - \dot{\mathbf{q}}_r \\ = & \dot{\mathbf{q}} - \mathbf{J}^+(\mathbf{q})(\dot{\mathbf{x}}_f - \bar{\mathbf{T}}^{-1} \dot{\mathbf{T}} \Delta \mathbf{x}) + \alpha \mathbf{J}^+(\mathbf{q}) \bar{\mathbf{T}}^{-1} \Delta \boldsymbol{\varepsilon}, \end{aligned} \quad (29)$$

and $\dot{\mathbf{q}}_r$ is a reference vector, $\mathbf{J}^+(\mathbf{q})$ denotes the pseudo-inverse of $\mathbf{J}(\mathbf{q})$, α is a positive constant, $\mathbf{K}_s$ is a diagonal and positive definite matrix, k_g is a positive constant, and $\text{sgn}(\cdot)$ represents the sign function.

In equation (28), $\boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f)$ is the vector of activation function, and $\hat{\mathbf{W}}_d$ is the estimated weight matrix of a RBF neural network that will be employed to approximate the dynamic model of the robotic manipulator. The estimated weight matrix $\hat{\mathbf{W}}_d$ is updated by the following update law:

$$\dot{\hat{\mathbf{W}}}_{dj}^T = -\mathbf{L}_{dj}^{-1} \boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f) s_j, \quad (30)$$

where $\hat{\mathbf{W}}_{dj}$ represents the j^{th} row vector of $\hat{\mathbf{W}}_d$, s_j is the j^{th} element of $\mathbf{s}$, and $\mathbf{L}_{dj}$ are diagonal positive definite matrices.

Using the sliding vector $\mathbf{s}$ defined by equation (29), the robot dynamics described by equation (2) is written as:

$$\begin{aligned} & \mathbf{M}(\mathbf{q}) \dot{\mathbf{s}} + [\tfrac{1}{2} \dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})] \mathbf{s} \\ & + \mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}_r + [\tfrac{1}{2} \dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})] \dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}, \end{aligned} \quad (31)$$

and the term $\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}_r + [\tfrac{1}{2} \dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})] \dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q})$ is approximated by the RBF neural network as:

$$\begin{aligned} & \mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}}_r + [\tfrac{1}{2} \dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})] \dot{\mathbf{q}}_r + \mathbf{g}(\mathbf{q}) \\ & = \mathbf{W}_d \boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f) + \mathbf{E}_d, \end{aligned} \quad (32)$$

where $\mathbf{W}_d$ is the ideal weight matrix, and $\mathbf{E}_d$ is the vector of approximation error. Substituting equations (28) and (32) into equation (31), we obtain the following closed-loop equation:

$$\begin{aligned} & \mathbf{M}(\mathbf{q}) \dot{\mathbf{s}} + [\tfrac{1}{2} \dot{\mathbf{M}}(\mathbf{q}) + \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})] \mathbf{s} + \mathbf{K}_s \mathbf{s} \\ & + \mathbf{J}^T(\mathbf{q}) \bar{\mathbf{T}}^T \Delta \boldsymbol{\varepsilon} + k_g \text{sgn}(\mathbf{s}) \\ & + \Delta \mathbf{W}_d \boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f) + \mathbf{E}_d = \mathbf{0}, \end{aligned} \quad (33)$$

where $\Delta \mathbf{W}_d = \mathbf{W}_d - \hat{\mathbf{W}}_d$, and $\Delta \mathbf{W}_d(0)$ represents the initialization error at $t=0$. The initial estimates of the weights $\hat{\mathbf{W}}_d(0)$ can be set as zero. Note that the neural-network compensation of the dynamics reduces to zero when the weights are set as zero when $t=0$, but the feedback term in the proposed controller will keep the tracking errors bounded. The algorithm for the proposed controller is given in Algorithm 1.

Algorithm 1 Adaptive NN Control with A Unified Bound

Specify the type of the bound according to robot tasks;
Specify the parameters of the bound according to Table I;
Initialize the weight of neural network $\hat{\mathbf{W}}_d(0)$;
Set the parameters k_g , $\mathbf{K}_s$, $\mathbf{L}_{dj}$, and α ;
for The task is not completed **do**
 Calculate the activation functions θ_i in equation (3);
 Calculate the sliding vector $\mathbf{s}$ in equation (29);
 Apply the control input $\boldsymbol{\tau}$ using equation (28);
 Update the estimated weight $\hat{\mathbf{W}}_d$ using equation (30);
end for

To prove the stability of the robot system, a Lyapunov-like candidate is defined as:

$$V = \frac{1}{2} \mathbf{s}^T \mathbf{M}(\mathbf{q}) \mathbf{s} + P_o(\Delta \mathbf{x}_T) + \frac{1}{2} \sum_{j=1}^n \Delta \mathbf{W}_{dj} \mathbf{L}_{dj} \Delta \mathbf{W}_{dj}^T. \quad (34)$$

Next, differentiating V with respect to time, we have:

$$\begin{aligned} \dot{V} = & \mathbf{s}^T \mathbf{M}(\mathbf{q}) \dot{\mathbf{s}} + \frac{1}{2} \mathbf{s}^T \dot{\mathbf{M}}(\mathbf{q}) \mathbf{s} + (\dot{\mathbf{T}} \Delta \mathbf{x} + \bar{\mathbf{T}} \Delta \dot{\mathbf{x}})^T \Delta \boldsymbol{\varepsilon} \\ & - \sum_{j=1}^n \dot{\hat{\mathbf{W}}}_{dj} \mathbf{L}_{dj} \Delta \mathbf{W}_{dj}^T. \end{aligned} \quad (35)$$

Substituting the closed-loop equation (33), the sliding vector in equation (29) and the update law in equation (30) into equation (35) and using the properties of robot dynamics, we have:

$$\begin{aligned} \dot{V} = & -\mathbf{s}^T \mathbf{K}_s \mathbf{s} - \mathbf{s}^T \{ \mathbf{J}^T(\mathbf{q}) \bar{\mathbf{T}}^T \Delta \boldsymbol{\varepsilon} + k_g \mathbf{s} \mathbf{g} \mathbf{n}(\mathbf{s}) \\ & + \Delta \mathbf{W}_d \boldsymbol{\theta}_d(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f) + \mathbf{E}_d \} \\ & + (\dot{\mathbf{T}} \Delta \mathbf{x} + \bar{\mathbf{T}} \Delta \dot{\mathbf{x}})^T \Delta \boldsymbol{\varepsilon} - \sum_{j=1}^n \dot{\hat{\mathbf{W}}}_{dj} \mathbf{L}_{dj} \Delta \mathbf{W}_{dj}^T \\ = & -\mathbf{s}^T \mathbf{K}_s \mathbf{s} - \mathbf{s}^T \mathbf{E}_d - k_g \mathbf{s}^T \mathbf{s} \mathbf{g} \mathbf{n}(\mathbf{s}) - \alpha \Delta \boldsymbol{\varepsilon}^T \Delta \boldsymbol{\varepsilon}. \end{aligned} \quad (36)$$

Note that $-\mathbf{s}^T \mathbf{E}_d - k_g \mathbf{s}^T \mathbf{s} \mathbf{g} \mathbf{n}(\mathbf{s}) \leq -(k_g - b_d) \|\mathbf{s}\|$ where b_d denotes the upper bound of $\mathbf{E}_d$. Therefore, if the control parameter k_g is set such that

$$k_g > b_d, \quad (37)$$

it is obtained that:

$$\dot{V} \leq -\mathbf{s}^T \mathbf{K}_s \mathbf{s} - \alpha \Delta \boldsymbol{\varepsilon}^T \Delta \boldsymbol{\varepsilon} \leq 0. \quad (38)$$

Since the error $\mathbf{E}_d$ for the ideal weight $\mathbf{W}_d$ can be reduced by choosing sufficient neurons [31], k_g is chosen sufficiently large to satisfy condition (37). We now state the following theorem.

Theorem 1: *When the proposed control scheme and the update law in equations (28) and (30) are applied to robotic manipulator, the closed-loop system is stable and gives rise to the convergence of the task-space position of the end effector to the unified bound, if the control parameter k_g is chosen*

sufficiently large such that condition (37) is satisfied.

Proof: If condition (37) is satisfied, $V > 0$ and $\dot{V} \leq 0$. Therefore, V is bounded. Since V is bounded, $\mathbf{s}$, $P_o(\Delta \mathbf{x}_T)$, and $\Delta \mathbf{W}_d$ are all bounded. The boundedness of $P_o(\Delta \mathbf{x}_T)$ ensures the boundedness of the functions $f_i(\Delta \mathbf{x}_T)$. Therefore, $\Delta \mathbf{x}$ is also bounded. Since $\mathbf{s}$, $\Delta \mathbf{x}$, and $\Delta \mathbf{W}_d$ are all bounded, the closed-loop system is stable.

In addition, the boundedness of $\Delta \mathbf{x}$ ensures the boundedness of $\mathbf{x}$ since $\mathbf{x}_f$ is bounded. Since the functions $f_i(\Delta \mathbf{x}_T)$ are specified as scalar functions with continuous partial derivatives, the boundedness of $\mathbf{x}$, $\Delta \mathbf{x}$ ensures the boundedness of the partial derivatives. Since both the functions of bound and its partial derivatives are bounded, the unified region error $\Delta \boldsymbol{\varepsilon}$ is bounded as seen from equation (27).

The boundedness of $\Delta \boldsymbol{\varepsilon}$ ensures the boundedness of $\dot{\mathbf{q}}_r$. Since both $\mathbf{s}$ and $\dot{\mathbf{q}}_r$ are bounded and $\mathbf{s} = \dot{\mathbf{q}} - \dot{\mathbf{q}}_r$, $\dot{\mathbf{q}}$ is bounded. The boundedness of $\dot{\mathbf{q}}$ also ensures the boundedness of $\dot{\mathbf{x}}$ since $\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q}) \dot{\mathbf{q}}$ and $\mathbf{J}(\mathbf{q})$ are trigonometric functions of $\mathbf{q}$. The boundedness of $\dot{\mathbf{x}}$ ensures the boundedness of the time derivative of the region error $\Delta \dot{\boldsymbol{\varepsilon}}$, which implies that $\Delta \boldsymbol{\varepsilon}$ is uniformly continuous. From equation (38), it is obtained that $\mathbf{s}, \Delta \boldsymbol{\varepsilon} \in L_2(0, +\infty)$ [3], [32]. Then it follows [3], [33], [34] such that:

$$\Delta \boldsymbol{\varepsilon} \rightarrow \mathbf{0}, \quad (39)$$

which implies that $\mathbf{f}(\Delta \mathbf{x}_T) \leq \mathbf{0}$ or $\mathbf{x} \rightarrow \mathbf{x}_f$. In either case, the end effector is inside the unified bound.

1) : When the unified bound is not specified as the performance bound, the end effector can start outside the bound during transient stage and it has been shown that the position of the end effector converges to the desired bound at steady state.

2) : When the unified bound is specified as the performance bound, the performance bound encloses the initial position error such that the initial region error $\Delta \boldsymbol{\varepsilon}(0) = \mathbf{0}$. Suppose that the end effector exceeds the performance bound at a time instant where $f_i(\Delta \mathbf{x}_T) > 0$, $\Delta \boldsymbol{\varepsilon}$ becomes arbitrarily large (see Fig. 7 and Fig. 8), which is contradicted with the conclusion that the region error $\Delta \boldsymbol{\varepsilon}$ is continuous and bounded throughout the robot movement. Therefore, the end effector always stays inside the performance bound. $\triangle \triangle \triangle$

Remark 2: Since the end effector starts inside the performance bound and stays within it, the performance bound could be set to ensure that the robot is away from the singular configurations so that $\mathbf{J}^+(\mathbf{q})$ is non-singular. $\triangle \triangle \triangle$

Remark 3: The proposed controller is different from the setpoint controller. In robot setpoint control, the controller requires specific positions of desired points, and hence each desired point has to be determined exactly. In this paper, the robot end effector is controlled to track a unified bound, and the position of the end effector converges to any point inside the bound and not necessarily the reference point. $\triangle \triangle \triangle$

V. EXPERIMENT

The proposed control scheme was implemented on the first two joints of a SCARA robot, where the lengths of the first and the second link are $l_1 = 0.35$ m, $l_2 = 0.33$ m respectively. The experimental setup [35] was shown in Fig. 9.

The joint motors of the robot are driven by servo amplifiers. The amplifiers are connected to a ServoToGo I/O card (model II). The servo I/O card is an ISA-bus based general purpose data acquisition card. The optical incremental encoders in the robot monitor the joint positions with a resolution of 500 lines. Joint velocities are obtained from differentiation of the joint angles. The 24-bit counters of the card are read by a computer serving as the controller in which one Pentium III 450 MHz processor and 128 MB DRAM are installed. The control signals are fed through the digital-to-analogue converters of the servo I/O card to the amplifiers. The digital-to-analog converters have a 13-bit resolution and the output voltage has a -10 volt to $+10$ volt range.

Fig. 9. Experimental setup

A. Moving Region

In the first experiment, a moving circular region was formulated as the desired position of robot end effector, and the region function in equation (5) was specified as:

$$f(\Delta \mathbf{x}_T) = \frac{\Delta x_{T1}^2}{0.05^2} + \frac{\Delta x_{T2}^2}{0.05^2} - 1 \leq 0, \quad (40)$$

and the transformation matrix was set as $\bar{T} = \mathbf{I}_2$ where $\mathbf{I}_2 \in \mathbb{R}^{2 \times 2}$ is an identity matrix, and hence the size of the region was constant. The end effector started from an initial position at $(-0.25, 0.09)$ m and tracked the moving circular region, where the trajectory of the reference position (x_{f1}, x_{f2}) was specified as:

$$\begin{cases} x_{f1} = -0.35 + 0.15\cos(0.4t) \text{ m} \\ x_{f2} = 0.10 + 0.15\sin(0.4t) \text{ m}. \end{cases} \quad (41)$$

The control parameters in equation (28) were set as: $\alpha = 1$, $k_{xi} = 1$, $\mathbf{K}_s = \text{diag}\{0.0002, 0.0002\}$. The input of the neural network was specified as: $\mathbf{p} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f]^T \in \mathbb{R}^{10}$. Ten neurons were utilized in the neural network, and the center and the width of the activation functions were set as: $\mu_{j1} = 1.104$, $\mu_{j2} = -1.057$, $\mu_{j3} = 0$, $\mu_{j4} = 0$, $\mu_{j5} = -0.2$, $\mu_{j6} = 0.1$, $\mu_{j7} = 0$, $\mu_{j8} = 0$, $\mu_{j9} = 0$, $\mu_{j10} = 0$ and $\sigma_j = 10$ where $j = 1, \dots, 10$. The parameters for the update law in equation (30) were set as: $\mathbf{L}_d = 0.0001\mathbf{I}_{10}$ where $\mathbf{I}_{10} \in \mathbb{R}^{10 \times 10}$ is an identity matrix, and the weights of neural network were initialized as $\bar{\mathbf{W}}_d(0) = \mathbf{0}$.

The path of the robot end effector and the tracking errors are shown in Fig. 10 and Fig. 11(a) respectively. As seen from Fig. 10 and Fig. 11(a), the end effector is able to track the moving circular region. The input torques are shown in

Fig. 11(b). In the experiment, the number of the neurons is sufficient to approximate the robot dynamics with negligible error, and hence the control term $-k_g \text{sgn}(\mathbf{s})$ is not required. That is, the control parameter k_g is set as zero.

B. Dynamic Region

In the second experiment, a dynamic region described by equation (5) was specified as the desired position of robot end effector as:

$$f(\Delta \mathbf{x}_T) = \frac{\Delta x_{T1}^2}{r^2} + \frac{\Delta x_{T2}^2}{r^2} - 1 \leq 0, \quad (42)$$

where $r = 0.6$ m represents the initial radius of the circular region, and the transformation matrix $\bar{T}$ was specified as:

$$\bar{T} = \begin{bmatrix} \frac{1}{0.03+e^{-0.4t}} & 0 \\ 0 & \frac{1}{0.03+e^{-0.4t}} \end{bmatrix}. \quad (43)$$

That is, the dynamic region is initially large and scaled down in the end. The end effector started from the initial position at $(-0.11, 0.64)$ m and tracked the shrinking region. The trajectory of the reference position (x_{f1}, x_{f2}) was now specified as a lemniscate of Bernoulli as:

$$\begin{cases} x_{f1} = -0.3 + \frac{0.1\cos(0.5t)}{1+\sin^2(0.5t)} \text{ m} \\ x_{f2} = 0.1 + \frac{0.1\sin(0.5t)\cos(0.5t)}{1+\sin^2(0.5t)} \text{ m}. \end{cases} \quad (44)$$

The control parameters in equation (28) were set as: $\alpha = 1$, $k_{xi} = 0.01$, and $\mathbf{K}_s = \text{diag}\{0.5, 0.5\}$. The input of the neural network was specified as: $\mathbf{p} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f]^T \in \mathbb{R}^{10}$. Ten neurons were utilized in the neural network, and the center and the width of the activation functions were set as: $\mu_{j1} = 1.104$, $\mu_{j2} = -1.057$, $\mu_{j3} = 0$, $\mu_{j4} = 0$, $\mu_{j5} = -0.2$, $\mu_{j6} = 0.1$, $\mu_{j7} = 0$, $\mu_{j8} = 0$, $\mu_{j9} = 0$, $\mu_{j10} = 0$ and $\sigma_j = 10$ where $j = 1, \dots, 10$. The parameters for the update law in equation (30) were set as: $\mathbf{L}_d = 0.0001\mathbf{I}_{10}$, and the weights of neural network were initialized as $\bar{\mathbf{W}}_d(0) = \mathbf{0}$.

The tracking errors are shown in Fig. 11(c). As seen from Fig. 11(c), the tracking errors are within the bounds of the dynamic region, and the position of end effector converges to the reference trajectory as the region is scaled down. The input torques are shown in Fig. 11(d). The input torques increase when the robot end effector is near the sharp curvature of Bernoulli trajectory, so as to keep the end effector following the dynamic region.

The path of the robot end effector is shown in Fig. 12. As seen from Fig. 12, the end effector starts from an initial position inside the dynamic region and stays within it. Initially, the region is large so that less control effort is required. As the size of the region is scaled down, the dynamic region reduces to the Bernoulli trajectory, and the position of end effector also converges to the desired trajectory.

Next, the reference position was specified as a setpoint as $\mathbf{x}_f = [-0.3, 0.1]^T$ m, while the control parameters remain the same. The end effector was controlled to start from the same initial position and track the shrinking region. The path of the robot end effector and the position errors are shown in Fig. 13. As the size of the region is scaled down, the dynamic region reduces to the setpoint, and the position of the end effector also converges to the desired point, as seen from Fig. 13.

Fig. 10. Experiment 1: the robot end effector is controlled to track the moving circular region.

Fig. 11. Region tracking control

Fig. 12. Experiment 2: the dynamic region is reduced to a Bernoulli trajectory.

Fig. 13. Experiment 2: the dynamic region is reduced to a setpoint.

C. Performance Bound

In the third experiment, the unified bound was specified as performance bound. First, a performance bound with an allowable maximum overshoot in equation (12) was specified as:

$$\begin{aligned} f_1(\Delta x_{T1}) &= \frac{\Delta x_{T1}^2}{b_1^2} - 1 \leq 0, \\ f_2(\Delta x_{T2}) &= \frac{\Delta x_{T2}^2}{b_2^2} - 1 \leq 0, \end{aligned} \quad (45)$$

where

$$\begin{aligned} b_1 &= \begin{cases} 0.3 \text{ m}, & x_1 \geq x_{f1}, \\ -0.1 \text{ m}, & x_1 < x_{f1}, \end{cases} \\ b_2 &= \begin{cases} 0.6 \text{ m}, & x_2 \geq x_{f2}, \\ -0.2 \text{ m}, & x_2 < x_{f2}, \end{cases} \end{aligned} \quad (46)$$

and the transformation matrix $\bar{T}$ was specified as:

$$\bar{T} = \begin{bmatrix} \frac{1}{0.04 + e^{-0.4t}} & 0 \\ 0 & \frac{1}{0.03 + e^{-0.4t}} \end{bmatrix}. \quad (47)$$

From equations (13) and (14), the performance of the transient response can be obtained as: the allowable maximum overshoot $M_p = \frac{1}{3}$, the convergence speed $l_c = 0.4$, and the steady-state error $e_{ss} = -0.004$ m in the coordinate of x_1 and $e_{ss} = -0.006$ m in the coordinate of x_2 .

Next, another performance bound without overshoot in equation (15) was specified as:

$$\begin{aligned} f_1(\Delta x_{T1}) &= \begin{bmatrix} \frac{\Delta x_{T1}^2}{0.3^2} - 1 \\ 1 - \frac{\Delta x_{T1}^2}{0.05^2} \end{bmatrix} \leq 0, \\ f_2(\Delta x_{T2}) &= \begin{bmatrix} \frac{\Delta x_{T2}^2}{0.75^2} - 1 \\ 1 - \frac{\Delta x_{T2}^2}{0.45^2} \end{bmatrix} \leq 0, \end{aligned} \quad (48)$$

and the transformation matrix was the same. Reference regions were introduced to reduce the possibility of oscillation. For the performance bound in equation (45), the reference region was specified as: $f_{r1}(\Delta x_{T1}) = \Delta x_{T1}^2 / b_{r1}^2 - 1 \leq 0$, and $f_{r2}(\Delta x_{T2}) = \Delta x_{T2}^2 / b_{r2}^2 - 1 \leq 0$, where $b_{r1} = 0.2$ m if $x_1 \geq x_{f1}$ else $b_{r1} = -0.01$ m, and $b_{r2} = 0.4$ m if $x_2 \geq x_{f2}$ else $b_{r2} = -0.02$ m. For the performance bound in equation (48), the reference region was specified as: $f_{r1}(\Delta x_{T1}) = [\frac{\Delta x_{T1}^2}{0.2^2} - 1, 1 - \frac{\Delta x_{T1}^2}{0.15^2}]^T \leq 0$, and $f_{r2}(\Delta x_{T2}) = [\frac{\Delta x_{T2}^2}{0.6^2} - 1, 1 - \frac{\Delta x_{T2}^2}{0.5^2}]^T \leq 0$.

The end effector started from the same initial position at $(-0.11, 0.64)$ m and stayed inside the performance bounds described by equations (45) and (48) respectively. The trajectory of the reference position (x_{f1}, x_{f2}) was set as:

$$\begin{cases} x_{f1} = -0.2 - 0.005t \text{ m} \\ x_{f2} = 0.3 - 0.005t \text{ m}. \end{cases} \quad (49)$$

When the performance bound in equation (45) was employed, the control parameters in equation (28) were set as: $\alpha = 1$, $k_{xi} = 100$, and $\mathbf{K}_s = \text{diag}\{0.7, 0.7\}$. When the performance bound in equation (48) was employed, the control parameters were set as: $\alpha = 1$, $k_{xi} = 100$, and $\mathbf{K}_s = \text{diag}\{1.5, 1.5\}$. The input of the RBF neural network was specified as: $\mathbf{p} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f]^T \in \mathbb{R}^{10}$. Ten neurons were utilized in the neural network, and the center and the width of the activation functions were set as: $\mu_{j1} = 1.104$, $\mu_{j2} = -1.057$, $\mu_{j3} = 0$, $\mu_{j4} = 0$, $\mu_{j5} = 0.18$, $\mu_{j6} = 0.3$, $\mu_{j7} = 0$, $\mu_{j8} = 0$,

$\mu_{j9} = 0$, $\mu_{j10} = 0$ and $\sigma_j = 10$ where $j = 1, \dots, 10$. The parameters for the update law in equation (30) were set as: $\mathbf{L}_d = 0.0001 \mathbf{I}_{10}$, and the weights of neural network were initialized as $\hat{\mathbf{W}}_d(0) = \mathbf{0}$.

The transient responses of the closed-loop system is shown in Fig. 14(a) and Fig. 14(c) respectively. As seen from Fig. 14(a) and Fig. 14(c), the end effector starts inside the performance bound and stays within it. As the bound shrinks, the tracking errors reduce accordingly. Note that the overshoots are eliminated in Fig. 14(a) and Fig. 14(c). This is because the transient response is guaranteed when the tracking errors are bounded inside the performance bound. The input torques are shown in Fig. 14(b) and Fig. 14(d). Similarly, the control parameter k_g is set as zero since the number of the neurons is sufficient to approximate the robot dynamics with negligible error.

D. Energy-Saving Property

In the proposed control method, the gradient of the potential energy remains zero when the end effector starts from an initial position within the unified bound and stays inside the bound throughout the robot movement. Comparatively, the position error $\Delta \mathbf{x}$ in the standard adaptive neural network controller is nonzero until the end effector reaches the reference position. Therefore, using the unified bound requires less energy than using the conventional position error.

In the fourth experiment, the energy-saving property of the proposed control method is verified. To monitor the energy consumption, the voltage across the DC motor was recorded by the ADC module in the ServoToGo card, and the current was measured by using the clamp-on AC/DC current probes (Fluke 80i-110s). The output of current probe was also recorded by the ADC module in the ServoToGo card. The total energy consumption for the robotic manipulator is computed as:

$$E = \int_0^t (v_1 i_1 + v_2 i_2) dt, \quad (50)$$

where v_1 and v_2 denote the voltage, and i_1 and i_2 represent the current across the DC motors of the first and second joints respectively.

First, the proposed controller was implemented in the robot system, and the robot end effector started from $(-0.11, 0.64)$ m inside the bound and stayed within it. The reference position (x_{f1}, x_{f2}) was set as a setpoint as: $(-0.2, 0.1)$ m. As the bound shrinks to a neighborhood around the reference position $\mathbf{x}_f$, the position of end effector converges to the reference position accordingly. The control parameters in equation (28) were set as: $\alpha = 1$, $k_{xi} = 30$, $\mathbf{K}_s = \text{diag}\{0.001, 0.001\}$.

Next, a standard adaptive neural network controller using the position error $\Delta \mathbf{x}$ was implemented in the robot system for comparison, which was specified as:

$$\begin{aligned} \boldsymbol{\tau}_c &= -k_p \mathbf{J}^T(\mathbf{q}) \Delta \mathbf{x} - \mathbf{K}_v \mathbf{s}_c \\ &+ \hat{\mathbf{W}}_c \boldsymbol{\theta}_c(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}_f, \dot{\mathbf{x}}_f, \ddot{\mathbf{x}}_f) - k_g \text{sgn}(\mathbf{s}_c), \end{aligned} \quad (51)$$

where $\mathbf{s}_c = \dot{\mathbf{q}} - \mathbf{J}^+(\mathbf{q}) \dot{\mathbf{x}}_f + \alpha \mathbf{J}^+(\mathbf{q}) \Delta \mathbf{x}$ is the sliding vector, and $\hat{\mathbf{W}}_c$ is the estimated weight matrix of the neural network,

(a) Transient response with performance bound in equation (45) (b) Input Torques with performance bound in equation (45) (c) Transient response with performance bound in equation (48) (d) Input Torques with performance bound in equation (48)

Fig. 14. Experiment 3: the transient response of the closed-loop system is guaranteed when the tracking errors are inside the performance bound.

TABLE II
COMPARISON OF ENERGY CONSUMPTION (J)

NO.	1	2	3	4	5	6	7	8	9	10	11	12	AVG.	DEV.
Proposed	0.954	0.938	0.938	0.82	0.944	0.93	0.942	0.95	0.936	0.862	0.956	0.84	0.918	0.002
Standard	1.328	1.284	1.248	1.176	1.288	1.272	1.272	1.286	1.246	1.202	1.271	1.2	1.258	0.002

and $\theta_c(q, \dot{q}, x_f, \dot{x}_f, \ddot{x}_f)$ is the activation function. The control parameters remain the same.

The energy consumption is computed based on equation (50), and Table II illustrates the quantitative comparisons based on the twelve experimental results, by using the two controllers in equations (28) and (51) respectively. The comparison of the energy accumulation in one experiment is shown in Fig. 15. As seen from Table II and Fig. 15, the proposed controller requires less energy than the standard controller.

Fig. 15. Experiment 4: it is seen that the proposed controller requires less energy as compared to the standard adaptive neural network controller.

VI. CONCLUSION

In this paper, a new adaptive neural network controller with a unified objective bound has been proposed for robotic manipulator. The proposed unified bound is a generalization of setpoint, trajectory and performance bound, and it can be specified as the dynamic region to suit different requirements of robot tasks. The stability of the closed-loop system is analyzed by using Lyapunov-like analysis, and experimental results are presented to illustrate the performance of the proposed approach and the energy-saving property of the proposed neural network controller with the dynamic region.

REFERENCES

- [1] D. Sun, S. Hu, X. Shao, and C. Liu, "Global stability of a saturated nonlinear PID controller for robot manipulators," *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 4, pp. 892-899, 2009.
- [2] Y. Karayiannidis, and Z. Doulgeri, "Model-free robot joint position regulation and tracking with prescribed performance guarantees," *Robotics and Autonomous Systems*, Vol. 60, No. 2, pp. 214-226, 2012.
- [3] S. Arimoto, *Control Theory of Nonlinear Mechanical Systems - A Passivity-Based and Circuit-Theoretic Approach*. Oxford, Clarendon.
- [4] J. J. E. Slotine, and W. Li, "On the adaptive control of robot manipulators," *International Journal of Robotics Research*, No. 6, pp. 49-59, 1987.
- [5] G. Niemeyer, and J. J. E. Slotine, "Performance in adaptive manipulator control," *International Journal of Robotics Research*, Vol. 10, No. 2, pp. 149-161, 1991.
- [6] H. Berghuis, R. Ortega, and H. Nijmeijer, "A robust adaptive robot controller," *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 6, pp. 825-830, 1993.
- [7] F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators*, New York, Macmillan, 1993.
- [8] P. Tomei, "Robust adaptive friction compensation for tracking control of robot manipulators," *IEEE Transactions on Automatic Control*, Vol. 45, No. 11, pp. 2164-2168, 2000.
- [9] C. C. Cheah, C. Liu, and J. J. E. Slotine, "Adaptive tracking controls for robots with unknown kinematics and dynamic properties," *International Journal of Robotics Research*, Vol. 25, No. 3, pp. 283-296, 2006.
- [10] C. C. Cheah, D. Q. Wang, and Y. C. Sun, "Region-reaching control of robots," *IEEE Transactions on Robotics*, Vol. 23, No. 6, pp. 1260-1264, 2007.
- [11] C. C. Cheah, S. P. Hou, and J. J. E. Slotine, "Region-based shape control for a swarm of robots," *Automatica*, Vol. 45, pp. 2406-2411, 2009.
- [12] S. P. Hou, and C. C. Cheah, "Can a simple control scheme work for a formation control of multiple autonomous underwater vehicles?" *IEEE Transactions on Control Systems Technology*, Vol. 19, No. 5, pp. 1090-1101, 2011.
- [13] R. M. Sanner, and J. J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks*, Vol. 3, No. 6, pp. 837-863, 1992.
- [14] R. M. Sanner, and J. J. E. Slotine, "Stable adaptive control of robot manipulators using "neural" networks," *Neural Computation*, Vol. 7, No. 4, pp. 753-790, 1995.
- [15] R. M. Sanner, and J. J. E. Slotine, "Structurally dynamic wavelet networks for adaptive control of robotic systems," *International Journal of Control*, Vol. 70, No. 3, pp. 405-421, 1998.
- [16] F. L. Lewis, K. Liu, and A. Yesildirek, "Neural net robot controller with guaranteed tracking performance," *IEEE Transactions on Neural Networks*, Vol. 6, No. 3, pp. 703-715, 1995.

- [17] F. L. Lewis, A. Yesildirek, and K. Liu, "Multilayer neural net robot controller with guaranteed tracking performance," *IEEE Transactions on Neural Networks*, Vol. 7, No. 2, pp. 388-399, 1996.
- [18] F. L. Lewis, "Neural network control of robot manipulators," *Intelligent System and Their Applications*, Vol. 11, No. 3, pp. 64-75, 1996.
- [19] S. S. Ge, T. H. Lee, and C. J. Harris, *Adaptive Neural Network Control of Robotic Manipulators*. World Scientific Publishing, 1998.
- [20] R. Carelli, E. F. Camacho, and D. Patino, "A neural network based feedforward adaptive controller for robots," *IEEE Transactions on Systems, Man, Cybernetics*, Vol. 25, No. 9, pp. 1281-1288, 1995.
- [21] H. D. Patino, R. Carelli, and B. R. Kuchen, "Neural networks for advanced control of robot manipulators," *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, pp. 343-354, 2002.
- [22] S. J. Yoo, J. B. Park, and Y. H. Choi, "Adaptive output feedback control of flexible-joint robots using neural networks: Dynamic surface design approach," *IEEE Transactions on Neural Networks*, Vol. 19, No. 10, pp. 1712-1726, 2008.
- [23] L. Cheng, Z. Hou, and M. Tan, "Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model," *Automatica*, Vol. 45, No. 10, pp. 2312-2318, 2009.
- [24] Y. Zhao, and C. C. Cheah, "Neural network control of multifingered robot hands using visual feedback," *IEEE Transactions on Neural Networks*, Vol. 20, No. 5, pp. 758-767, 2009.
- [25] S. Bhasin, K. Dupree, P. M. Patre, and W. E. Dixon, "Neural network control of a robot interacting with an uncertain viscoelastic environment," *IEEE Transactions on Control Systems Technology*, Vol. 19, No. 4, pp. 947-955, 2011.
- [26] L. Wang, T. Chai, and C. Yang, "Neural-network-based contouring control for robotic manipulators in operational space," *IEEE Transactions on Control Systems Technology*, Vol. 20, No. 4, pp. 1073-1080, 2012.
- [27] C. Bechlioulis, Z. Doulgeri, and G. Rovithakis, "Neuro-adaptive force/position control with prescribed performance and guaranteed contact maintenance," *IEEE Transactions on Neural Networks*, Vol. 21, No. 12, pp. 1857-1868, 2010.
- [28] C. P. Bechlioulis, and G. A. Rovithakis, "Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance," *IEEE Trans. Automatic Control*, Vol. 53, No. 9, pp. 2090-2099, 2008.
- [29] C. P. Bechlioulis, and G. A. Rovithakis, "Adaptive control with guaranteed transient and steady state tracking error bounds for strict feedback systems," *Automatica*, Vol. 45, No. 2, pp. 532-538, 2009.
- [30] C. P. Bechlioulis, and G. A. Rovithakis, "Prescribed performance adaptive control for multi-input multi-output affine in the control nonlinear systems," *IEEE Trans. Automatic Control*, Vol. 55, No. 5, pp. 1220-1226, 2010.
- [31] J. A. Farrell, and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*. New York: John Wiley & Sons, 2006.
- [32] C. A. Desoer, and M. Vidyasagar, *Feedback Systems: Input-Output Properties*. New York: Academic Press, 1975.
- [33] M. W. Spong, and M. Vidyasagar, *Robot Dynamics and Control*. New York: John Wiley & Sons, 1989.
- [34] J. J. E. Slotine, and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, New Jersey: Prentice Hall, 1991.
- [35] C. C. Cheah, "Task-space PD control of robot manipulators: unified analysis and duality property", *International Journal of Robotics Research*, Vol. 27, No. 10, pp. 1152-1170, 2008.

Chien Chern Cheah (SM'03) was born in Singapore. He received B.Eng. degree in Electrical Engineering from National University of Singapore in 1990, M.Eng. and Ph.D. degrees in Electrical Engineering, both from Nanyang Technological University, Singapore, in 1993 and 1996, respectively.

From 1990 to 1991, he worked as a design engineer in Chartered Electronics Industries, Singapore. He was a research fellow in the Department of Robotics, Ritsumeikan University, Japan from 1996 to 1998. He joined the School of Electrical and Electronic Engineering, Nanyang Technological University as an assistant professor in 1998. Since 2003, he has been an associate professor in Nanyang Technological University. In November 2002, he received the overseas attachment fellowship from the Agency for Science, Technology and Research (A*STAR), Singapore to visit the Nonlinear Systems laboratory, Massachusetts Institute of Technology. He serves as an associate editor for *IEEE Transactions on Robotics*, *Automatica* and *Asian Journal of Control*.

Xiang Li (M'12) received the Bachelor's, Master's and Ph.D. degrees from Beijing Institute of Technology, Beijing, China, and Nanyang Technological University, Singapore, in 2006, 2008, and 2013, respectively.

He is currently a Research Fellow with the Intelligent Robotics Lab, Nanyang Technological University. His current research interests include robot control, visual servoing, and cell manipulation.

Dr. Li has served as a Reviewer for several international journals including *Automatica*, the *IEEE*

Transactions on Robotics, the *IEEE Transactions on Mechatronics*, and the *Asian Journal of Control*. He received the Highly Commended Paper Award at the Third IFToMM International Symposium on Robotics and Mechatronics.