International Journal of Control, Automation and Systems 15(2) (2017) 952-958
http://dx.doi.org/10.1007/s12555-015-0070-7
ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Stability Analysis of Reference Compensation Technique for Controlling Robot Manipulators by Neural Network

Seul Jung

Abstract: Neural network control for robot manipulators is aimed to compensate for uncertainties in the robot dynamics. The location of a compensating point differentiates the control scheme into two categories, the feedback error learning (FEL) scheme and the reference compensation technique (RCT). The RCT scheme is relatively less used although it has several structural advantages. In this paper, the global stability of the RCT scheme is analyzed on the basis of Lyapunov function. The analysis turns out that the stability depends upon the magnitude of the controller gains. Simulation studies of controlling the position of a two-link robot manipulator are conducted.

Keywords: Neural network control, RCT scheme, robot manipulators, stability.

1. INTRODUCTION

The terminology of 'Intelligence' is used everywhere to humanize the system by mimicking human brain functions to deal with various applications. Intelligent robots or intelligent systems are one of leading areas of applying intelligence to the machines.

To apply intelligence to the system, appropriate intelligent tools such as neural network, fuzzy logic, genetic algorithms or others are selected to satisfy performance specifications.

For the real-time control applications, neural networks and fuzzy logic are used most among them. Fuzzy logic is a powerful tool to transfer semantic human expression to numerical machine expression. Although determining optimal fuzzy rules requires a time consuming process, fuzzy logic can be duplicated with ease on the embedded system once rules are found. This is the reason why many fuzzy-related products are on the market.

Neural networks are a massive parallel computing structure that mimics human brain functions. Neural networks have the capabilities of mapping any nonlinear functions, of learning and adapting the environment, and of generalizing the data if input and output data are given. No rules are adjusted, but internal weights are automatically learned.

Neural network control for robot manipulators has been one of active research areas in robotics, control, and intelligent system communities. A simple idea of replacing a feedforward controller with a neural network controller provides on-line learning and control capability. This scheme is called feedback error learning (FEL) control [1]. In the literature, a majority of neural network control applications uses this scheme because the stability of initial learning stage can be guaranteed by feedback controllers and the inverse of the system can eventually be learned at the convergence.

In the framework of the FEL scheme, similar control schemes along with the stability analysis have been proposed. A neural network controller is designed for controlling a 5 DOF robot manipulator [2]. Stability analysis of a RBF neural network control scheme has been done for a robot manipulator [3]. Neural network control for multiple robotic manipulators has been presented [4]. Neural network control schemes along with sliding mode control or back stepping control have been presented [5,6]. Stability analysis of the FEL scheme for robot manipulator control has been extensively presented by reformulating robot dynamic equation with tracking error functions along with simulation results [7].

In a meanwhile, the modification of the control structure by moving the compensating position of a neural network yields several structural advantages. Since the neural network compensates at the trajectory level, this scheme is called the reference compensation technique (RCT) [8,9]. Although the RCT scheme has been successfully applied to real robot systems, the stability of the RCT controlled system has not been addressed while the stability of FEL scheme has been addressed heavily in the literature [10].

Therefore, in this paper, the stability of RCT scheme based on the Lyapunov sense for controlling position of

Manuscript received February 11, 2015; revised September 23, 2015, December 15, 2015, February 3, 2016, and May 23, 2016; accepted June 4, 2016. Recommended by Associate Editor Xiaojie Su under the direction of Editor Fuchun Sun. This work has been supported in part by the 2014 National Research Foundation of Korea (NRF-2014R1A2A1A11049503).

Seul Jung is with the Intelligent Systems and Emotional Engineering (ISEE) Laboratory, Department of Mechatronics Engineering, Chungnam National University, 99 Daehak-ro, Yuseong-gu, Daejeon 34134, Korea (e-mail: jungs@cnu.ac.kr).

robot manipulators is analyzed. Simulation studies are conducted.

2. ROBOT MANIPULATOR DYNAMICS

The dynamic equation of an n degrees-of-freedom robot manipulator in the joint space coordinates is given by

$$D(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)=\tau, \tag{1}$$

where the vectors q is the $n\times 1$ joint angle, $\dot{q}$ is the $n\times 1$ joint angular velocity, and $\ddot{q}$ is the $n\times 1$ joint angular acceleration, $D(q)$ is the $n\times n$ symmetric positive definite inertia matrix, $C(q,\dot{q})\dot{q}$ is the $n\times 1$ Coriolis and centrifugal torque vector, $G(q)$ is the $n\times 1$ gravitational torque vector, and τ is the $n\times 1$ vector of actuator joint torque vector.

Define the trajectory tracking errors as

$$e=q_d-q,\quad \dot{e}=\dot{q}_d-\dot{q},\quad \ddot{e}=\ddot{q}_d-\ddot{q}, \tag{2}$$

where q_d is the reference trajectory. Based on (2), we define the error surface function as

$$s=\dot{e}+\lambda e,\quad \dot{s}=\ddot{e}+\lambda\dot{e}. \tag{3}$$

Rearranging $\dot{q}$, $\ddot{q}$ from (3) in terms of s, $\dot{s}$ yields

$$\dot{q}=\dot{q}_d-(s-\lambda e),\quad \ddot{q}=\ddot{q}_d-(\dot{s}-\lambda\dot{e}). \tag{4}$$

Substituting $\dot{q}$, $\ddot{q}$ in (4) into (1) yields the modified dynamic equation of robot manipulators in terms of s [7].

$$f(q,\dot{q})-(D\dot{s}+Cs)=\tau \tag{5}$$

where $f(q,\dot{q})=D(q)(\ddot{q}_d+\lambda\dot{e})+C(\dot{q}_d+\lambda e)+G(q)$ which includes all the uncertainties. This term is expected to be cancelled out through the compensation process by a neural network. As a result, equation (5) satisfies the stability with ease since the closed loop equation becomes the function of the error s when the controller is designed with the function of the error s.

3. NEURAL NETWORK STRUCTURE

Here a multilayered perceptron network with linear output structure is used. The neural network has one hidden layer and one output layer as shown in Fig. 1. The nonlinear function of the hidden layer is given as the sigmoidal function.

The output of the hidden unit is defined as

$$\psi_j(x_i)=\frac{1-\exp(-\sum_{i=0}^{N_I}x_i)}{1+\exp(-\sum_{i=0}^{N_I}x_i)}, \tag{6}$$

where x_i is the ith input element and N_I is the number of input ellements.

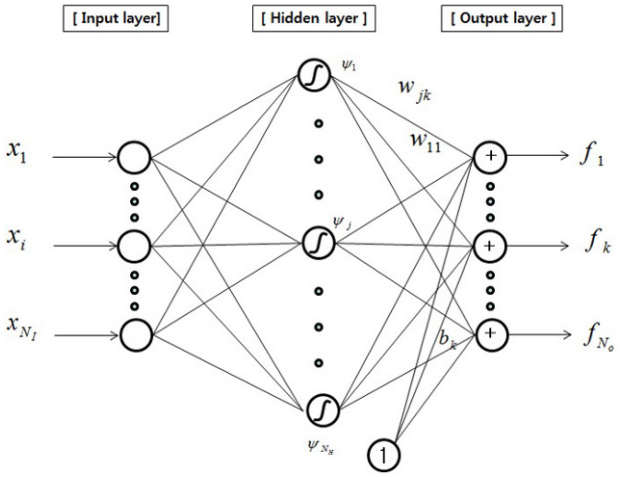


Fig. 1. Neural network structure.

The output is summed together as a linear function.

$$f_k=\sum_{j=1}^{N_H}\psi_j w_{jk}+b_k, \tag{7}$$

where ψ_j is jth output of the hidden layer and w_{jk} is the weight between the jth hidden unit and the kth output, b_k is the bias weight of the kth output, and $N_H$is the number of hidden units.

4. REFERENCE COMPENSATION TECHNIQUE

4.1. PD control structure

The RCT scheme has been known as an input shaping nonlinear controller that uses a neural network as an auxiliary controller while a feedback controller is considered as a main controller. Neural network outputs are compensated at the desired trajectory level. Initial transient stability is guaranteed by the feedback controller and the final steady state is governed by a neural network controller.

The RCT scheme provides some structural advantages. Since neural network compensation is done at the reference input level, the feedback controller cannot be modified while FEL requires the modification of the feedback controller when neural network control is implemented to control systems. Neural compensation can be done through wireless communication so that unmanned aircrafts like drones can be controlled remotely from the ground.

Consider a PD controlled robot system as shown in Fig. 2. The PD controller output becomes

$$\begin{aligned}\tau_C&=K_D\dot{e}+K_P e\\&=K(\dot{e}+\lambda e)\\&=Ks,\end{aligned} \tag{8}$$

where K is the controller gain such as $K=K_D$.

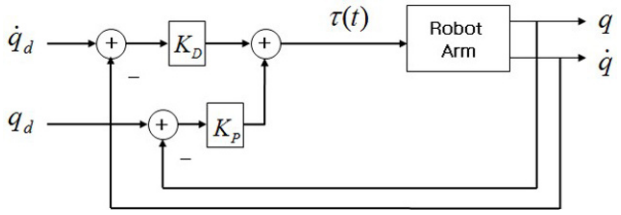


Fig. 2. PD control structure.

Combining (8) with (5) yields the closed error equation as

$$D\dot{s} + (C+K)s = f(q,\dot{q}). \tag{9}$$

To satisfy the tracking error $s = 0$, the function $f(q,\dot{q})$ should be zero. Neural network is used for this purpose.

Neural network is required to compensates for the function $f(q,\dot{q})$ as

$$D\dot{s} + (C+K)s = f(q,\dot{q}) - \Phi, \tag{10}$$

where Φ is the neural network output.

4.2. RCT scheme 1

Neural network outputs are added to the reference trajectories as shown in Fig. 3. The addition of the compensating signals from the neural network forms the closed error equation as below.

$$\begin{aligned}\tau &= K(\dot{q}_d - \dot{q} + \dot{q}_n + \lambda(q_d - q + q_n)) \\ &= K(\dot{e} + \lambda e) + K(\dot{q}_n + \lambda q_n) \\ &= K(s + \Phi_I),\end{aligned} \tag{11}$$

where q_n, $\dot{q}_n$ are neural network outputs and $\Phi_I = \dot{q}_n + \lambda q_n$.

Combining (11) with (5) yields the closed loop error equation as

$$D\dot{s} + (C+K)s = f(q,\dot{q}) - K\Phi_I, \tag{12}$$

where Φ is the neural network output.

If neural network output cancels out the function $f(q,\dot{q})$, then (12) becomes

$$D\dot{s} + (C+K)s = 0. \tag{13}$$

Define the training signals v as

$$v = K(\dot{e} + \lambda e) = Ks. \tag{14}$$

Then (12) becomes

$$\begin{aligned}v &= \tau - K(\dot{q}_n + \lambda q_n) \\ &= \tau - K\Phi_I.\end{aligned} \tag{15}$$

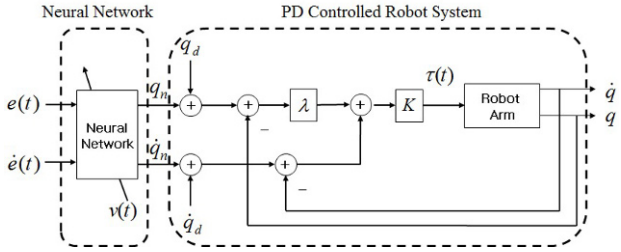


Fig. 3. RCT control structure I.

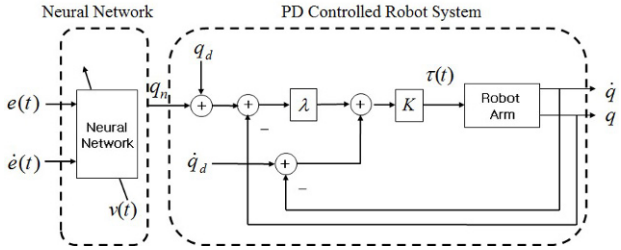


Fig. 4. RCT control structure II.

4.3. RCT scheme 2

A simplified RCT structure is shown in Fig. 4. Neural network output is added to the joint angle only. Addition of the compensating signals to the PD controlled system yields the closed error equation.

$$\begin{aligned}\tau &= K(\dot{q}_d - \dot{q} + \lambda(q_d - q + q_n)) \\ &= K(\dot{e} + \lambda e) + K\lambda q_n \\ &= K(s + \Phi_{II}),\end{aligned} \tag{16}$$

where $\Phi_{II} = \lambda q_n$.

Combining (16) with (5) yields the closed loop error equation as

$$D\dot{s} + (C+K)s = f(q,\dot{q}) - K\Phi_{II}. \tag{17}$$

Then (17) becomes

$$\begin{aligned}v &= \tau - K\lambda q_n \\ &= \tau - K\Phi_{II}.\end{aligned} \tag{18}$$

4.4. Learning process

Learning of the neural network uses the back-propagation algorithm to update weights. Since we need to minimize the feedback control error v instead of e, the objective function to be minimized is formed as

$$E = \frac{1}{2} v^T v, \tag{19}$$

where $v \in R^{n\times 1}$ is considered as the training signal.

The gradient with respect to the weight is used and calculated to minimize the objective function as

$$\Delta w = -\eta \frac{\partial E}{\partial w} = -\eta \frac{\partial E}{\partial v}\frac{\partial v}{\partial w}, \tag{20}$$

where η is the learning rate and w is the weight vector. The gradient can be obtained from (15) and (18) as

$$\Delta w = -\eta \frac{\partial v}{\partial w} v = \eta [\frac{\partial \Phi}{\partial w}]^T K v, \tag{21}$$

where Φ can be either Φ_I or Φ_{II}.

Weights are updated at every sampling time as

$$w(t+1) = w(t) + \Delta w(t) + \alpha \Delta w(t-1), \tag{22}$$

where α is the momentum constant.

5. STABILITY ANALYSIS

The universal approximation property of neural network gives the function as

$$f(q, \dot{q}) = K W^T \psi + \varepsilon, \tag{23}$$

where W is the weight matrix, ψ is the output vector of the hidden layer, ε is the approximation error, and K is the constant diagonal matrix.

From (11) and (16) the control law becomes the multiplication of gain K to the addition of the feedback controller and the neural network output.

$$\tau = K(\hat{W}^T \psi + s). \tag{24}$$

Substituting (23) and (24) into (5) yields the closed loop error equation.

$$D\dot{s} + (C+K)s = K\tilde{W}^T \psi + \varepsilon, \tag{25}$$

where $\tilde{W}^T = W^T - \hat{W}^T$ and W^T is the true weights, $\hat{W}^T$ is the approximation of the weight.

Define the Lyapunov function as

$$L = \frac{1}{2} s^T D s + \frac{1}{2} Tr\{\tilde{W}^T \Gamma^{-1} \tilde{W}\}, \tag{26}$$

where Tr is Trace of the matrix.

Differentiating (26) yields

$$\dot{L} = \frac{1}{2} s^T \dot{D} s + s^T D \dot{s} + Tr\{\tilde{W}^T \Gamma^{-1} \dot{\tilde{W}}\}. \tag{27}$$

From (25)

$$D\dot{s} = K\tilde{W}^T \psi - (C+K)s + \varepsilon. \tag{28}$$

Substituting $D\dot{s}$ of (28) into (27) yields

$$\begin{aligned} \dot{L} = & \frac{1}{2} s^T (\dot{D} - 2C) s - s^T K s \\ & + Tr\{\tilde{W}^T (\Gamma^{-1} \dot{\tilde{W}} + K \psi s^T)\} + s^T \varepsilon. \end{aligned} \tag{29}$$

Using the skew symmetry property of robot manipulator $\dot{D} - 2C = 0$ simplifies (29) as

$$\dot{L} = -s^T K s + Tr\{\tilde{W}^T (\Gamma^{-1} \dot{\tilde{W}} + K \psi s^T)\} + s^T \varepsilon. \tag{30}$$

To guarantee the stability, we select the update law for the weights as

$$\dot{\tilde{W}} = \dot{W}^* - \dot{\hat{W}} = -\dot{\hat{W}} = -\Gamma K \psi s^T. \tag{31}$$

Substituting the update law $\dot{\hat{W}} = \Gamma K \psi s^T$ into (30) yields

$$\dot{L} = -s^T K s + s^T \varepsilon. \tag{32}$$

To satisfy $\dot{L} < 0$ for the stability, we can select the controller gain to satisfy the following condition.

$$|K| > \frac{|\varepsilon|}{|s|}. \tag{33}$$

Let us compare the update equations of (21) and (31). For simplicity, consider the RCT scheme II of which neural network output is compensated at the position only, not at the velocity level.

From (21), the gradient of $\frac{\partial \Phi_{II}}{\partial W}$ can be obtained as

$$\frac{\partial \Phi_{II}}{\partial W} = \lambda \frac{\partial q_n}{\partial W}. \tag{34}$$

Since the neural network output $q_n$is the summation, it is described as

$$q_n = W^T \psi. \tag{35}$$

The gradient of (35) with respect to W becomes

$$\frac{\partial q_n}{\partial W} = \psi. \tag{36}$$

Therefore, the update equation becomes

$$\frac{\partial \Phi_{II}}{\partial W} = \lambda \psi. \tag{37}$$

Therefore, the update equation (21) can be described as

$$\begin{aligned} \Delta W &= \eta \lambda K \psi v \\ &= \Gamma K \psi s^T, \end{aligned} \tag{38}$$

where $v = s^T$ and $\Gamma = \eta \lambda$. Ultimately (38) is same as (31).

6. SIMULATION STUDIES

6.1. Simulation setup

A two-link robot manipulator is used for the simulation studies. The mass of each link is 5 Kg and its length is 0.4 m. Initial joints are set to $[0.7854, -0.7854]$ as shown in Fig. 5. The robot is commanded to follow the desired trajectories such as sinusoidal motions at each joint.

The following friction term is added to each joint as an uncertainty of the robot system.

$$f = sign(q) * (k_1 * abs(q) + k_2), \tag{39}$$

where $k_1, k_2$are friction coefficients.

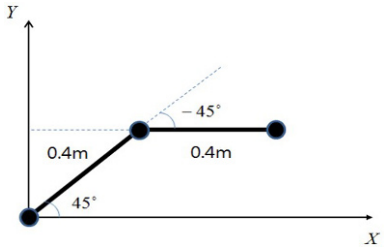


Fig. 5. Initial robot configuration.

6.2. PD control

PD gains are set to 100, 20 for K_P and K_D, respectively to give the critically damped response. PD controller gains are selected to have the critical response for the second order system. Given the same PD controller gains, our goal is to see the performance by a neural network controller.

Fig. 6 shows the tracking performance by PD controllers. Notable tracking errors due to uncertainties in robot dynamics are observed. Because of the coupled effect, the error of the joint 1 is larger than that of joint 2. The tacking error can be minimized with the model based control method.

6.3. Neural network control

Next experiment is to use the neural network structure in Fig. 1. Neural network has 6 hidden units. The learning rate of 0.00005 is tested and compared. Scheme II is used. Fig. 8 shows the tracking performance.

Fig. 8 shows the comparison plot between the proportional control error and a neural network output before controller gains are multiplied. Since the neural network output is added to the reference trajectory to compensate for the error, P error is close to zero after 0.5 seconds. We also note that the pattern of the compensating signal is similar to the tracking error by the PD controller shown in Fig. 6(b), which means that the neural network compensates for the tracking error.

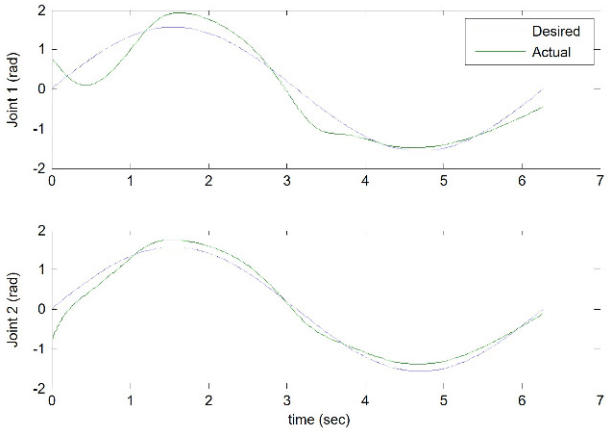
6.4. Effects of unknown payloads

Since it is worth to know the performance of the RCT control method for the effect of the dynamic parameter change, 5kg of a payload is added to the end-effector. Fig. 9 shows the tracking responses for the PD control and the RCT control method. We clearly see the better tracking performance by the RCT control method.

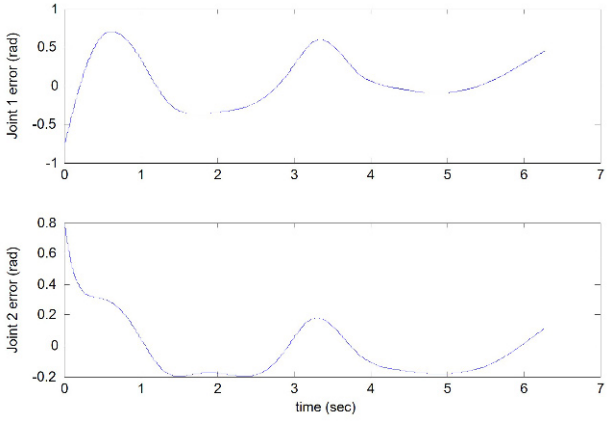
6.5. Effects of learning rate

The learning rate is quite a sensitive parameter to the stability although the global stability is guaranteed. At first, the learning rate is selected as a small value such as 0.000001 and then the learning rate is increased to maximize the tracking performance.

We have learned that the larger learning rate provides the better tracking performance. It is sensitive that the



(a) Tracking performance.



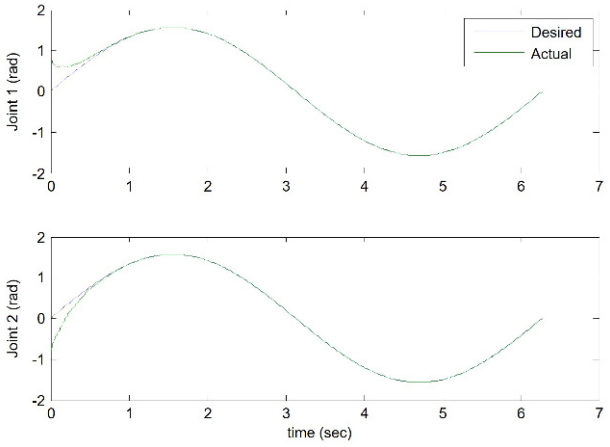
(b) Tracking error.

Fig. 6. PD control performance ($K_P = 100$, $K_d = 20$).

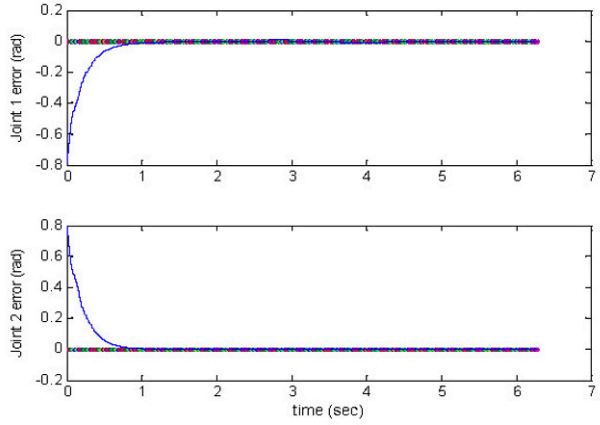
performance depends upon the learning rate. The leaning rate is selected as large as possible. However, we cannot keep increasing the learning rate. The learning rate has a limit to be increased for the stable performance. For instance, when $\eta = 0.0001$ is used, the system becomes easily unstable.

6.6. Comparison with feedback error learning control method

Next simulation is to compare with the feedback error learning (FEL) control method. The major difference between RCT and FEL is the compensating location of neural network in the control loop. Fig. 10 shows the tracking performances of RCT and FEL which are very compatible. The corresponding neural network outputs are plotted in Fig. 11. We see that the output of FEL is much larger than that of RCT. However, when the controller gain Kp (100) is multiplied to the neural network output of RCT scheme, they are almost similar in magnitude as shown in the bottom plot of Fig. 11.



(a) Tracking performance.



(b) Tracking error.

Fig. 7. NN control performance.

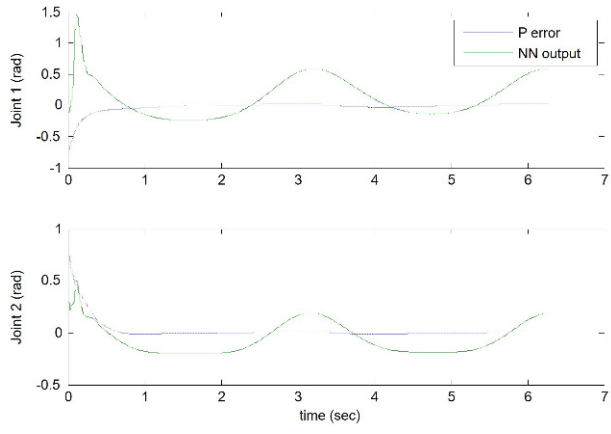


Fig. 8. NN compensating signal.

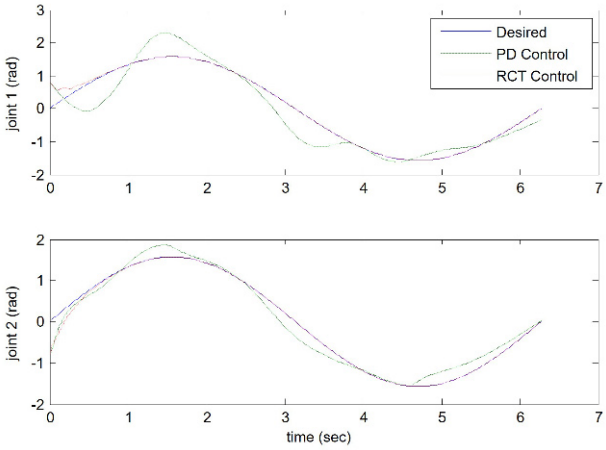


Fig. 9. Tracking performance for a 5 kg payload.

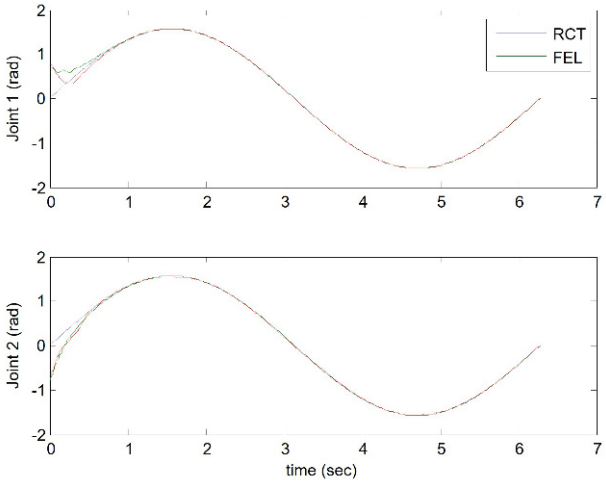


Fig. 10. Tracking performances by RCT and FEL.

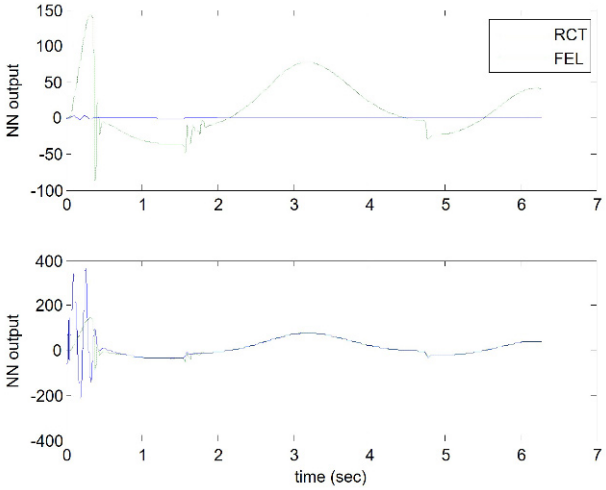


Fig. 11. NN outputs of RCT and FEL.

7. CONCLUSION

The RBF-like network is used as a neural controller for controlling the position of a robot manipulator. Neural network compensates for uncertainties at the trajectory level. Stability of the RCT neural network control scheme was analyzed in terms of energy function. It turns out that the RCT scheme is comparable to the FEL scheme. But the RCT scheme shows the structural advantages. Stability analysis shows that only difference between FEL and RCT scheme is the controller gain K, which can be included in the learning rate.

REFERENCES

[1] H. Gomi and M. Kawato, "Learning control for a closed loop system using feedback error learning," *Proc. of the IEEE International Conf. on Decision and Control*, pp. 3289-3294, 1990.

[2] X. Xie, L. Cheng, Z. Hou, and C. Ji, "Adaptive neural network control of a 5 DOF robot manipulator," *Proc. of Int. Conf. on Intelligent Control and Information Processing*, pp. 376-381, 2010.

[3] Y. Wu, Q. He, and C. Wang, "Adaptive neural learning control of rigid-link electrically-driven robot manipulators," *Proceedings of the 30th Chinese Control Conference*, pp. 6616-6622, 2011.

[4] D. Zhao, Q. Zhu, N. Li, and S. Li, "Neural network based synchronized control for multiple robotic manipulators," *Proc. of IEEE Conf. on Control and Automation*, pp. 1950-1955, 2013.

[5] R. J. Wai and R. Muthusamy, "Fuzzy-neural network inherited sliding-mode control for robot manipulator including actuator dynamics," *IEEE Trans. on Neural Network and Learning Systems*, vol. 24, no. 2, pp. 274-287, 2013.

[6] R. J. Wai and R. Muthusamy, "Design of fuzzy-neural-network-inherited backstepping control for robot manipulator including actuator dynamics," *IEEE Trans. on Fuzzy Systems*, vol. 22, no. 4, pp. 709-722, 2014.

[7] F. L. Lewis, S. Jagannathan, and A Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*, Taylor & Francis, 1999.

[8] S. Jung and T. C. Hsia, "Neural network inverse control techniques for PD controlled robot manipulator," *Robotica*, vol. 19, no. 3, pp. 305-314, 2000.

[9] S. Jung and H. T. Cho, "Decoupled neural network reference compensation technique for a PD controlled two degrees of freedom inverted pendulum," *International Journal of Control, Automation, and Systems Engineering*, vol. 2, no. 1, pp. 92-99, 2004.

[10] S. S. Ge and C. Wang, "Direct adaptive NN control of a class of nonlinear systems," *IEEE Trans. on Neural Networks*, vol. 13, no. 1, pp. 214-221, 2002.