Learning Feed-Forward Control for a Two-Link Rigid Robot Arm

Nguyen Duy Cuong
Electronics Faculty, Thai Nguyen University of Technology, Thai Nguyen City, Vietnam
Email: nguyenduycuong@tnut.edu.vn

Tran Xuan Minh
Electrical Faculty, Thai Nguyen University of Technology, Thai Nguyen City, Vietnam
Email: tranxuanminh@tnut.edu.vn

***Abstract*—This paper introduces a control structure which consists of a Proportional Derivative (PD) controller and a Neural Network (NN)-based Learning Feed-Forward Controller (LFFC) to a Two-Link Rigid Robot Arm. An on-line B-spline neural network is used because of its local weight-updating characteristic, which has the advantages of fast convergence speed and low computation complexity. The torque applied to each link is defined using the Euler-Lagrange equation. The controller design takes into account the troubles caused by inertial loading, coupling reaction forces between joints, and gravity loading effects. This control structure can be directly applied to different robots within the same class with different lengths and masses. Simulation results are presented to demonstrate the robustness of our proposed controller under serve changes of the system parameters.**

***Index Terms*—neural network (NN), learning feed-forward control (LFFC), two-link rigid robot arm**

I. Introduction

Two-link manipulators are two-degree-of-freedom robots [1]. They are major components in the manufacturing industry due to their several advantages including speed, accuracy, and repeatability. However, control of two-link manipulators to track accurately a given trajectory is an extremely challenging due to the dynamics is highly non-linear and complex [1]. The inherent non-linear and complex behavior the two-link manipulator system makes it difficult to achieve high level performance and robustness simultaneously.

Proportional-Integral-Derivative (PID) control is the most common control algorithm used in industrial control systems [2]. However, a PID controller has some limitations. In order to cope with the fact that the parameters of the system being controlled are slowly time-varying and/or uncertain, and the presence of reproducible disturbances, high PID gains are required. However, having large gains can lead to system instability. In general, for the PID control design, a compromise has to be made between performance and robust stability.

Manuscript received April 15, 2014; revised July 16, 2014.

In order to eliminate positional inaccuracy due to reproducible disturbances and model uncertainty we consider a learning feed-forward controller structure that consists of a feedback and a feed-forward controller [3]. We assume that the state of the process and the state of the reference model are identical and use the approximated inverse dynamics of the process to compute the feed-forward signal. For proper reference signals and when there are no disturbances, if the feed-forward controller equals the inverse of the plant, the tracking error will be zero. The feed-back controller is designed such that robust stability is guaranteed in the presence of model uncertainty, while the feed-forward controller is used to compensate for known reproducible disturbances.

Neural network literature for function approximation is sufficiently wealthy. In its complete form, the problem involves both parametric and structural learning [4], [5]. The properties of a neural network are determined by its structure and the learning rule is used to adapt the weights. The network structure is defined based on the basic processing units and the way in which they are interconnected. The network allows any unit to be connected to any other unit in the network. The principal importance of a neural network is not only the way a neuron is implemented but also how interconnections are made. The learning rules, also known as training algorithms, are procedures for modifying the weights on the connection links in a neural network. Finding appropriate learning rules and finding an appropriate network structure are two different issues. Neural networks typically take a vector of input values and produce a vector of output values. Inside, the weights of neurons are trained to build a representation of a function that will approximate the input to output mapping [5].

The aim of this paper is the design of the neural network based feed-forward controller for a two – link rigid robot arm. The problem that we are going to solve is to find a suitable feed-forward structure and parameters that allow achieving an as small as possible tracking errors with respect to the reference inputs with plant parameter variations.

doi: 10.12720/ijeee.3.4.279-284

This paper is organized as follows. Learning feed-forward control using B-spline neural network is introduced in Section II. In Section III, the dynamics of a two-link rigid robot arm is shown. The design of the proposed controller is introduced in Section IV. At the end of this paper, summary of the paper is given.

II. Learning Feed-Forward Control Using B-Spline Neural Network

A. Plant Disturbance Compensation

Motion control systems are usually designed to track given trajectories with small tracking errors. The assumed model of a plant with state dependent effects is illustrated in Fig. 1 [5].

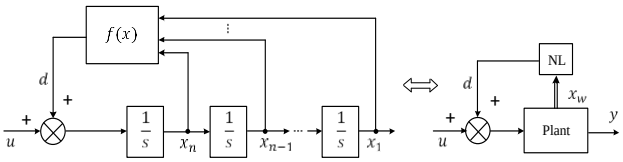

Figure 1. Plant and reproducible disturbances.

There are only integrators in the forward path. The state dependent effects are assumed to be known and act on the input that generates effects which are known as reproducible disturbances. Thus, reproducible disturbances can be viewed as a function of the state of the plant. These disturbances may significantly affect the control performances during the object operations.

B. Learning Feed-Forward Control Using B-Spline Neural Network

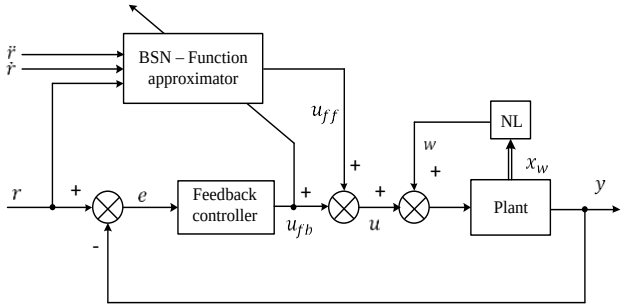

Figure 2. Learning feed-forward control; A B-spline neural network is used in the feed-forward controller.

The control structure of combination between feedback controller and learning feed-forward controller using B-spline neural network is shown in Fig. 2. This adopts direct inverse control of B-spline neural network. The neural network inverse model of the plant is in series with the plant such that the transfer function between the input of the system and the output of the plant is equal to 1. In order to obtain the input-output behavior of an inverse dynamics model over time, the B-spline network should be trained. During training, the plant output is compared with a desired reference output. The error between these two signals is used to adjust the weights which are known as the free parameters of the function approximator [5].

C. B-Spline Neural Network

The B-spline neural network (BSN) has become popular, mainly due to its ability to universally approximate any continuous nonlinear function [4], [5]. In a BSN the input-output mapping is created using basis functions, known as B-splines. A B-spline of order n consists of piece-wise polynomial functions of order $n-1$ (Fig. 3). The function evaluation of a B-spline is generally called the membership and is denoted as $\mu \in [0,1]$. That part of the input space for which $\mu \neq 0$ is known as its support. The output of the BSN is a weighted sum of the B-spline evaluations. We will only consider B-splines of order 2. In order to produce an input-output mapping from r, $\dot{r}$ and $\ddot{r}$ to y, the B-splines are placed on the domain of the BSN input such that at each input value the sum of all memberships equals 1. The output of the BSN is determined as

$$y(x) = \sum_{i=1}^{N} \omega_i \mu_i(x) \quad (1)$$

In which $y(x)$ is the output of the BSN, $\mu_i(x)$ is the membership of the $i-th$ basic function at input x, ω_i the weight of the basis function, and N is the number of B-splines. Training of the network can either be done after each sampling interval, which is known as on-line learning, or after a motion has been completed, known as off-line learning. In the on-line case, the cost function j that is minimized is the squared approximation error between the desired output of the BSN y_d and the actual output y:

$$J = \frac{1}{2}(y_d - y)^2 \quad (2)$$

Applying the back propagation rule, this yields:

$$\Delta\omega_i = -\gamma \frac{\partial J}{\partial \omega_i} = \gamma(y_d - y)\frac{\partial y}{\partial \omega_i} \quad (3)$$

Substituting (1) in (3) gives:

$$\Delta\omega_i = \gamma(y_d - y)\frac{\partial \sum_{j=1}^{N} \mu_j(x)\omega_j}{\partial \omega_i} = \gamma(y_d - y)\mu_i(x) \quad (4)$$

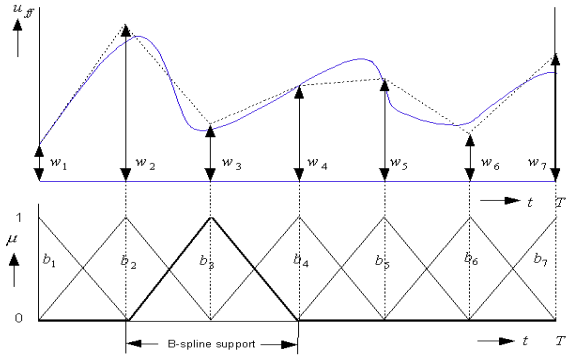

Figure 3. Top: I/O mapping; Bottom: 2^{nd} order B-splines.

In this equation $\Delta\omega_i$ denotes the adaptation of the weight of the $i-th$ B-spline, and γ the learning rate $0 < \gamma < 1$. In the off-line case the cost function is given by:

$$J = \frac{1}{2}\sum_j (y_{dj} - y_j)^2 \quad (5)$$

where y_{dj} is the desired output for input x_i and y_j the actual output of the BSN. The adaptation of weights is then determined as:

$$\Delta\omega_i = \gamma \sum_j (y_{d,j} - y_j)\mu_i^n(x_j) \quad (6)$$

In order to prevent large weight-adaptations, (6) is normalized by:

$$\Delta\omega_i = \gamma \frac{\sum_j (y_{dj} - y_j)\mu_i^n(x_j)}{\sum_j \mu_i^n(x_j)} \quad (7)$$

The off-line learning rule (7) implies that the weights are updated after a reference motion has been completed.

D. Design of LFFC Using BSNs

In the detailed design of LFFC the following parameters of the feed-forward part have to be considered based on stability analysis in the frequency domain [5]:

1) Selection of the inputs of the feed-forward part

The inputs of the BSN depend on the nature of the plant and the reproducible disturbances that the LFFC has to compensate. For random motions, the inputs should consist of the reference position and its derivatives/integrals.

2) Selection of the B-spline distribution on each of the inputs of the BSN

The BSN is able to accurately approximate a given signal by choosing a suitable width of the B-splines. A large number of "narrow" B-splines, allows the BSN to precisely approximate a high frequency signal. While a low frequency target signal requires a small number of "wide" B-splines.

3) The learning mechanism

Adaptation of the network weights can either be done after each sample, which is known as on-line learning, or after a motion has been completed, known as off-line learning.

4) The learning rate

The learning rate γ determines how strong the weights of the BSN are adjusted. A large learning rate implies that the weights are adjusted quickly. When a small learning rate is used the weights are adjusted slowly. However, a too high learning rate will cause un-stability.

By using a BSN, we have the following advantages:

a. Local learning: Due to the compact support of the B-splines, only a small number of the weights contribute to the output. Only these weights need to be updated during training. While, in the case of the Multi-Layer Perceptron (MLP), during training all network weights need to be updated. This means, the BSN converges much faster than the MLP.

b. No local minima: The output of the BSN is a linear function of the weights. This implies that, both for off-line and on-line learning the learning mechanisms do not suffer from local minima. This means that the initial weights of the BSN do not influence the final tracking accuracy.

c. Tune-able precision: The choice of the number of the B-splines and their locations determines the smoothness of the input-output relation. Random target signals can be accurately approximated, by choosing B-splines that have a proper compact support.

The principal drawback of BSNs is that the number of network weights grows exponentially with the dimension of the input space.

E. Input Selection of the LFFC

Models of the plant dynamics and of the reproducible disturbances are needed to create a structure for non-repetitive motions. We assume that the process can be modeled by a linear state space model and additive non-linear dynamics, shown in Fig. 4. This yields for the linear process

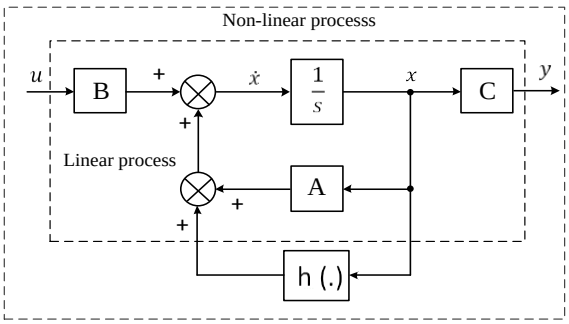

Figure 4. A non-linear process consists of a linear process a non-linearity $h(.)$

$$\dot{x} = Ax + Bu \quad (8)$$

The state vector is chosen such that it consists of positions (x_2) and their corresponding velocities (x_1), $(\dot{x}_2 = x_1)$. This gives

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}A_{11} & A_{12}\\ I & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}B_1\\ 0\end{bmatrix}u \quad (9)$$

Now we consider additive non-linear dynamics. For example, the case that the process has both a velocity and a position dependent non-linearity, this yields for the non-linear process:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}A_{11} & A_{12}\\ I & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}h(x_1) + h(x_2)\\ 0\end{bmatrix} + \begin{bmatrix}B_1\\ 0\end{bmatrix}u \quad (10)$$

The desired feed-forward signal u_d, that makes the actual position of the process x_2 tracks the desired statesx_{2d}, is given by (if B_1^{-1} exists):

$$B_1^{-1}\left(\ddot{x}_{2d} - A_{11}\dot{x}_{2d} - A_{12}x_{2d} - h(.)\right) = u_d \quad (11)$$

The specific inputs of a BSN can be determined on the basis of a (non-linear) state space representation of the plant dynamics. To ensure the LFFC to accurately control the process for random motions, the states involved in (11) should be used as inputs of the BSNs. In general, this will result in a BSN that has multiple inputs. It is clear that when the disturbances depend on the velocity of the plant, the reference velocity must be used as input of the LFFC. In case the disturbances do not only depend on the velocity of the plant, but also on the acceleration, both the reference velocity and acceleration must be used as inputs of the LFFC. Each of the reference inputs of the LFFC will be used to compensate one specific disturbance [3].

III. Dynamics Of Two-Link Rigid Robot ARM

To determine the dynamics, examine Fig. 5, where we have assumed that the link masses are concentrated at the ends of the links.

According to Lagrange's equation, the arm dynamics are given by the two coupled non-linear differential equations [6], [7]:

$$\tau_1 = [(m_1+m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2cos\theta_2]\ddot{\theta}_1 + [m_2l_2^2 + m_2l_1l_2cos\theta_2]\ddot{\theta}_2 - 2m_2l_1l_2\dot{\theta}_1\dot{\theta}_2sin\theta_2 - m_2l_1l_2\dot{\theta}_2^2sin\theta_2 + (m_1+m_2)gl_1cos\theta_1 + m_2gl_2\cos(\theta_1+\theta_2) \quad (12)$$

$$\tau_2 = [m_2l_2^2 + m_2l_1l_2cos\theta_2]\ddot{\theta}_1 + m_2l_2^2\ddot{\theta}_2 + m_2l_1l_2\dot{\theta}_1^2sin\theta_2 + m_2gl_2cos(\theta_1+\theta_2) \quad (13)$$

where θ_1, θ_2 angels of link 1, 2; m_1, m_2: masses of link 1, 2; a_1, a_2: lengths of link 1, 2.

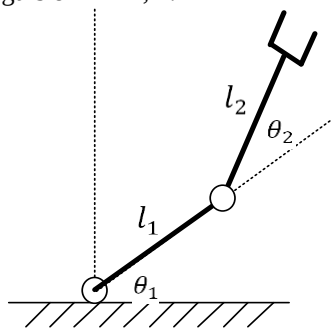

Figure 5. 2-DOF robot manipulator

Writing the arm dynamics in vector form yields

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \quad (14)$$

with

$$M_{11} = (m_1+m_2)l_1^2 + m_2l_2^2 + 2m_2l_1l_2cos\theta_2 \quad (15)$$

$$M_{12} = m_2l_2^2 + m_2l_1l_2cos\theta_2 \quad (16)$$

$$M_{21} = m_2l_2^2 + m_2l_1l_2cos\theta_2 \quad (17)$$

$$M_{22} = m_2l_2^2 \quad (18)$$

$$C_{11} = -2m_2l_1l_2\dot{\theta}_2sin\theta_2 \quad (19)$$

$$C_{12} = -m_2l_1l_2\dot{\theta}_2sin\theta_2 \quad (20)$$

$$C_{21} = m_2l_1l_2\dot{\theta}_1sin\theta_2 \quad (21)$$

$$C_{22} = 0 \quad (22)$$

$$G_1 = (m_1+m_2)gl_1cos\theta_1 + m_2gl_2cos(\theta_1+\theta_2) \quad (23)$$

$$G_2 = m_2gl_2cos(\theta_1+\theta_2) \quad (24)$$

These manipulator dynamics are in the standard form

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) = \tau \quad (25)$$

where θ is the vector of joint angels, $M(\theta)$ is the inertia-matrix, $C(\theta,\dot{\theta})$ describes Coriolis and centrifugal effects, $G(\theta)$ models gravitational load and τ is the vector of joint motor torques.

IV. Design Of Proposed Controller

A. Design the Reference Model

Explicit position, velocity, and acceleration profile setpoint signals are created using the reference model, which is described by the transfer function [8], [9]

$$H_{ref}(S) = \frac{\omega_n^2}{s^2+2\gamma\omega_n s+\omega_n^2} \quad (26)$$

The parameters of the reference model are chosen such that the higher order dynamics of the system will not be activated. The following numerical values are chosen:

$$\omega_n = 50, \gamma = 2 \quad (27)$$

B. Design the Feedback Controller

As discussed in the previous sections, tracking performance is obtained by the MRAS-based LFFC. The feedback controller is designed such that it features robust stability for closed loop when used alone. The compensator that is used is of the PD-type:

$$C(s) = K_p + K_ds \quad (28)$$

C. Determine the Inputs of the Feed-Forward Part

The inputs of the Feed –Forward components depend on the nature of the plant and the reproducible disturbances that the LFFC has to compensate. For random motions, the inputs should consist of the reference position and its derivatives/integrals. We may conclude that the joint angels of a rigid robot manipulator can be controlled by mean of LFFC. The desired feed-forward signal is:

$$M(\theta_d)\ddot{\theta}_d + C(\theta_d,\dot{\theta}_d)\dot{\theta}_d + G(\theta_d) = \tau_d \quad (29)$$

A modified version of the feedback PD control law:

$$\underbrace{M(\theta_d)\ddot{\theta}_d + C(\theta_d,\dot{\theta}_d)\dot{\theta}_d + G(\theta_d)}_{u_{feed-forward}} + \underbrace{K_pe + K_d\dot{e}}_{u_{feedback}} = \tau_d \quad (30)$$

where $e^T = [e_1 \quad e_2]$, $\theta^T = [\theta_1 \quad \theta_2]$, (31)

$K_p = [K_{p1} \quad 0; \quad 0 \quad K_{p2}]$, $K_d = [K_{d1} \quad 0; \quad 0 \quad K_{d2}]$, $\theta_d^T = [\theta_{1d} \quad \theta_{2d}]$, $e_1 = \theta_{1d} - \theta_1$, $e_2 = \theta_{2d} - \theta_2$, θ_d is the vector of desired joint angels, and θ is the vector of real joint angels. Substituting (13) in (31) yields the following desired feed-forward signals:

$$\begin{bmatrix} M_{11d} & M_{12d} \\ M_{21d} & M_{22d} \end{bmatrix}\begin{bmatrix} \ddot{\theta}_{1d} \\ \ddot{\theta}_{2d} \end{bmatrix} + \begin{bmatrix} C_{11d} & C_{12d} \\ C_{21d} & C_{22d} \end{bmatrix}\begin{bmatrix} \dot{\theta}_{1d} \\ \dot{\theta}_{2d} \end{bmatrix} + \begin{bmatrix} G_{1d} \\ G_{2d} \end{bmatrix} + \begin{bmatrix} K_{p1} & K_{d1} \\ 0 & 0 \end{bmatrix}\begin{bmatrix} e_1 \\ \dot{e}_1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ K_{p2} & K_{d2} \end{bmatrix}\begin{bmatrix} e_2 \\ \dot{e}_2 \end{bmatrix} = \begin{bmatrix} \tau_{1d} \\ \tau_{2d} \end{bmatrix} \quad (32)$$

From Eq. (12) and Eq. (13) we should select $[\theta_{1d}, \theta_{2d}, \dot{\theta}_{1d}, \dot{\theta}_{2d}, \ddot{\theta}_{1d}, \ddot{\theta}_{1d}, \sin(\theta_{2d}), \cos(\theta_{1d}), \cos(\theta_{1d}+\theta_{2d})]$as inputs of the feed-forward controller (see Table I and Fig. 6).

TABLE I. FEED-FORWARD COMPONENTS WITH CORRESPONDING INPUTS AND THE TARGET FUNCTIONS THAT THEY HAVE TO LEARN.

NO	INPUTS	LS	TARGET FUNCTIONS
FF_{11}	$cos\theta_{2d}\ddot{\theta}_{1d}$	u_1	$[(m_1+m_2)l_1^2+m_2l_2^2 + 2m_2l_1l_2cos\theta_2]\ddot{\theta}_1$
FF_{12}	$cos\theta_{2d}\ddot{\theta}_{2d}$	u_1	$[m_2l_2^2+m_2l_1l_2cos\theta_2]\ddot{\theta}_2$
FF_{13}	$sin\theta_{2d}\dot{\theta}_{1d}\dot{\theta}_{2d}$	u_1	$m_2l_1l_2sin\theta_2\dot{\theta}_1\dot{\theta}_2$
FF_{14}	$sin\theta_{2d}\ddot{\theta}_{2d}^2$	u_1	$m_2l_1l_2sin\theta_2\ddot{\theta}_2^2$
FF_{15}	$cos\theta_{1d}$	u_1	$(m_1+m_2)gl_1cos\theta_1$
FF_{16}	$cos(\theta_{1d}+\theta_{2d})$	u_1	$m_2gl_2cos(\theta_1+\theta_2)$
FF_{21}	$cos\theta_{2d}\ddot{\theta}_{1d}$	u_2	$[m_2l_2^2+m_2l_1l_2cos\theta_2]\ddot{\theta}_1$
FF_{22}	$\ddot{\theta}_{2d}$	u_2	$m_2l_2^2\ddot{\theta}_2$
FF_{23}	$sin\theta_{2d}\dot{\theta}_{1d}^2$	u_2	$m_2l_1l_2\dot{\theta}_1^2sin\theta_2$
FF_{24}	$cos(\theta_{1d}+\theta_{2d})$	u_2	$m_2gl_2cos(\theta_1+\theta_2)$

D. Choose the B-Spline Distribution

The aim of the BSN is to create the control signal that would result in the smallest possible tracking error. The width or support of the B-splines determines the accuracy such that the BSN is able to approximate an input-output relation. In case a one-dimensional BSN with second-order B-splines is used, the output consists of first-order polynomials between the B-spline knots. Therefore the smaller the distance between the B-spline knots the more accurate the LFFC can be obtained. When a small width is chosen the BSN is able to accurately approximate high frequency signals. Prior knowledge of the process and the disturbances are needed to determine the width of the B-splines. The minimum width of the B-splines d_{min} for which the system remains stable, can be determined on the basis of a Bode plot of the negative complementary sensitivity function, $|-T(j\omega)|_\infty$, of the closed loop. The following steps are given [3], [5]:

1. Determine $|-T(j\omega)|_\infty$ from a model of the closed loop system.

2. Use the Bode plot of the model to find out

$$\min_{\{\omega\epsilon R|\cos(\varphi)\le 0\}}|-T(j\omega)| \quad (33)$$

3. Search for the smallest value of ω_1 at which $\varphi_1 = \arg(-T(j\omega_1))$ satisfies

$$\varphi_1 = \arccos\left[-0.0147.\frac{|-T(j\omega)|_\infty}{\min_{\{\omega\epsilon R|cos(\varphi)\le 0\}}|-T(j\omega)|}\right] \quad (34)$$

4. The minimum value of the width of the B-splines,d_{min} is given by

$$d_{min} = \frac{2\pi}{\omega_1}[sec] \quad (35)$$

Because $-T(j\omega)$ depends on the feed-back controller, it follows that if the obtained value d_{min} is too large, the feedback controller should be redesigned.

E. Choose the Learning Rate

The learning rate determines how fast the weights of the BSN are adjusted. The larger the learning rate the stronger weights can be updated, while a small learning rate implies that the weights are updated slowly. The learning rate is chosen relatively small, for example $\gamma = 0.1$. Also, a large γ means less attenuation of noise.

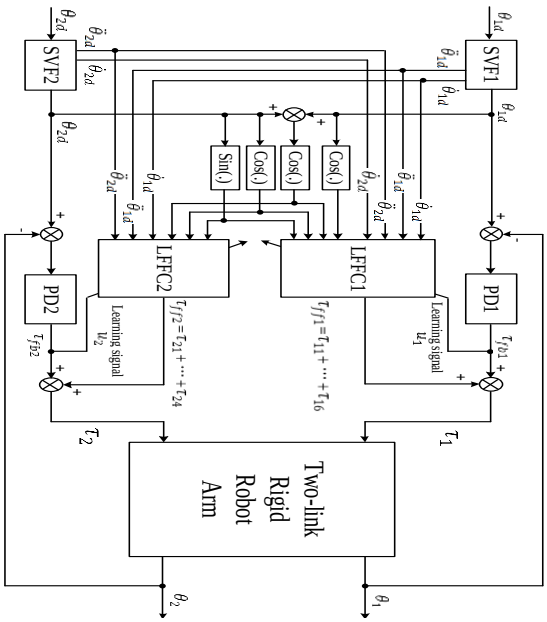

Figure 6. The proposed control structure

Simulation results

The parameter values of the considered two-link robotic arm are as follows: $m_1 = 0.5[kg]$; $m_2 = 0.5[kg]$; $l_1 = 0.4[m]$; $l_2 = 0.3[m]$; $g = 9.8\,{}^{m}/_{s^2}$; $K_p = [8000 \quad 0;\ 0 \quad 8000]$, $K_d = [50 \quad 0;\ 0 \quad 50]$. The desired tracking trajectories are supported to be: $\theta_{1d} = 10\sin(2\pi t)$ and $\theta_{2d} = 15\cos(1.2\pi t)$ for Link 1 and Link 2, respectively.

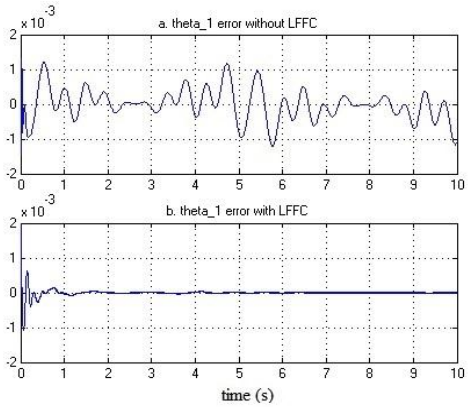

Figure 7. Comparison of the tracking error for theta_1 without (a) and with (b) LFFC.

Fig. 7.a and Fig. 8.a show the tracking errors using the PD controllers only. The control system is clearly stable. However, the tracking performance is not satisfactory. Fig. 7.b and Fig. 8.b show the tracking errors when LFFC controllers are added. Non-linear signals are well compensated (see Fig. 9 and Fig. 10). Obviously, the combination between PD controller and Neural Network based Learning Feed-Forward controller indeed

eliminates the tracking errors. In the beginning, the PD controller dominates. The LFFC controllers take less than haft second to learn and control the two – link robot motion very well.

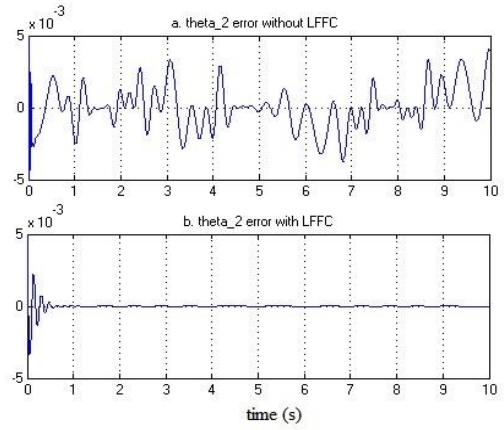

Figure 8. Comparison of the tracking error for theta_2 without (a) and with (b) LFFC.

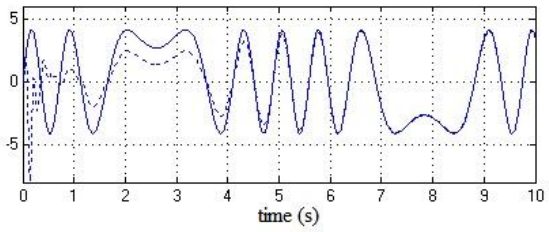

Figure 9. True non-linear signal $u_{15} = (m_1 + m_2)gl_1 cos\theta_1$(solid line) is well compensated by estimated signal (dashed line).

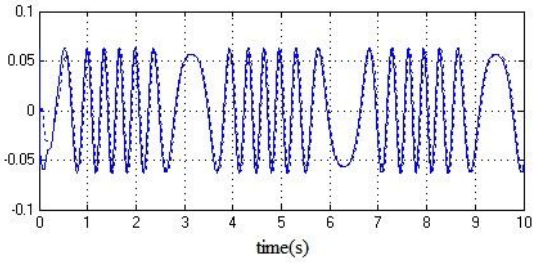

Figure 10. True non-linear coupling signal $\tau_{14} = m_2 l_1 l_2 sin\theta_2 \ddot{\theta}_2^2$ (solid line) is well compensated by estimated signal (dashed line).

V. Conclusion

The characteristics of the Learning Control introduced here are that the controller can be chosen as the inverse dynamical model of the plant and it is composed of the feed-forward loop besides the feedback controller loop. This type of learning control is known as the learning feed-forward control scheme. The feedback controller can be designed for robustness mainly, which does not require an accurate process model. The feed-forward controller compensates the reproducible disturbances that depend on the state of the process. Normally, a feed-forward controller is designed on the basis of an accurate process model of the process. The obtained LFFC does not suffer from a trade-off between high performance and robust stability.

Reference

[1] J. J. Craig, *Introduction to Robotics, Mechanics and Control*, 2nd Ed. Addison-Wesley, 1989.
[2] K. Ogata, *Modern Control Engineering*, Third Edition, Prentice Hall International, Simon and Schuster/A Viacom Company, Upper Saddle River, 1997.
[3] De Vries, J. A. Theo, L. J. Idema, and Velthuis, "Parsimonious Learning Feed-Forward Control," *European Symposium on Artificial Neural Networks ESANN*, 1998.
[4] De Vries, J. A. Theo, Velthuis and L. J. Idema, "Application of Parsimonious Learning Feed-Forward Control to Mechatronic Systems," *IEE Pro. Control Theory Appl*, Vol. 148, No. 4, July 2001.
[5] W. J. R. Velthuis, "Learning Feed-Forward Control: Theory, Design, and Applications," PhD thesis, University of Twente, Enschede, The Netherlands, 2000.
[6] G. R. Yu and L. W. Huang, "Design of LMI-Based Fuzzy Controller for Two-link Robot Arm using Genetic Algorithms," in *Proc. of 2008 CACS International Automatic Control Conference National Cheng Kung University*, Tainan, Taiwan, Nov. 21-23, 2008.
[7] R. Burkan, "Modeling of bound estimation laws and robust controllers for robustness to parametric uncertainty for control of robot manipulators", *Journal of Intelligent and Robotic Systems*, Vol. 60, pp. 365–394, 2011.
[8] N. D. Cuong, T. J. A. De Vries, and J. Van Amerongen, "Application of NN-based and MRAS-based FFC to electromechanical motion systems," in *Proc. EPE'13 ECCE Europe 15th European Conference on Power Electronics and Application*, 2013, pp. 1-10.
[9] D. C. Nguyen, "Advanced Controllers for Electromechanical Motion Systems," PhD thesis, University of Twente, Enschede, The Netherlands, 2008.

Nguyen Duy Cuong received the M.S. degree in Electrical Engineering from the Thai Nguyen University of Technology, Thai Nguyen city, Viet Nam, in 2001, the Ph.D. degree from the University of Twente, Enschede City, the Netherlands, in 2008.He is currently a lecturer with Electronics Faculty, Thai Nguyen University of Technology, Thai Nguyen City, Vietnam. His current research interests include real-time control, linear, parameter-varying systems, and applications in the industry. Dr. Nguyen DuyCuong has held visiting positions with the Universityat Buffalo- the State Universityof New York (USA) in 2009 and with the Oklahoma State University (USA) in 2012.

Tran Xuan Minh received the M.S. degree, and the Ph.D degree in Automation and Control Engineering from the Ha Noi University of Science and Technology, Viet Nam, in 1997 and in 2008, respectively.
He is currently a lecturer with Electrical Faculty, Thai Nguyen University of Technology, Thai Nguyen City, Viet Nam. His current research interests include electronics Power, Process Control, and applications in the industry. Dr. Tran Xuan Minh has held visiting positions with the Oklahoma State University (USA) in 2013.