A Hybrid CKF-NNPID Controller for MIMO Nonlinear Control System

Adna Sento[1] and **Yuttana Kitjaidure**[2], Non-members

ABSTRACT

This paper presents a detailed study to demonstrate the online tuning dynamic neural network PID controller to improve a joint angle position output performance of 4- joint robotic arm. The proposed controller uses a new updating weight rule model of the neural network architecture using multi-loop calculation of the fusion of the gradient algorithm with the cubature Kalman filter (CKF) which can optimize the internal predicted state of the updated weights to improve the proposed controller performances, called a Hybrid CKF-NNPID controller. To evaluate the proposed controller performances, the demonstration by the Matlab simulation program is used to implement the proposed controller that connects to the 4-joint robotic arm system. In the experimental result, it shows that the proposed controller is a superior control method comparing with the other prior controllers even though the system is under the loading criteria, the proposed controller still potentially tracks the error and gives the best performances.

Keywords: Robotic Arm Control System, Neural Network PID Controller, Cubature Kalman Filters, Adaptive Learning Algorithm

1. INTRODUCTION

Nowadays, the conventional PID controller might be replaced with intelligent controller in case of the uncertain parameters of the system models and complexity of the mathematical models and especially in the nonlinear systems with disturbances. In addition, human body interface controlling the robot is recently popular which can apply in the impaired human devices such as robot arm-assisted human impaired, robot-assisted human movement, robotic therapy on the recovery in chronic stroke, etc. Consequently, these have motivated the researchers to develop the system of control, particularly in the improvement of the part planning which may use either of the inverse/forward kinematics or computational intelligence. However, these methods still require the precision of the errors of the robot angle movements. Furthermore, the best responses of the robot movement will make it reached the robot target effectively. As a result, a large number of the intelligent control techniques have been devised to improve the controller for sophisticated angle precision control problems in the fusion of dynamic neural network with the classical control such a conventional PID controller, also known as neural PID controller (NNPID controller). For example, S. Cong and Y. Liang [1] proposed the neural PID controller to improve the performance of the multi-variable control system that used a resilient back propagation neural network algorithm for updating weight rule. In 2010, Ho Phanm Huy Anh [2] provided the neural PID controller which used the back propagation algorithm to update the weight of the neural network. Wen Yu and R. Jacob [3] also proposed the Neural PID controller to the upper limb exoskeleton robot. Furthermore, Vikas Kumar also proposed the neural PID controller to control the position of the permanent magnet synchronous motor (PMSM) [4]. Recently, F. G. Rossomando and C. M. Soria proposed the neural PID controller to control the mobile robot [5]. They used the gradient algorithm to update the weight of the controller. In addition, there are other examples of the fusion of the computational intelligence with the conventional PID controller that were established such as [6], [7], and [8].

The basic idea of the conventional PID is usually known by three-term of the controller as given by the general discrete function, $u(tk)$, as follows:

$$u(t_k) = K_p e(t_k) + K_l \sum_{i=1}^{k} e(t_i)\Delta t + K_D \frac{e(t_k)-e(t_{k-1})}{\Delta t} \quad (1)$$

where K_P, K_I, and K_D are called the controller gains which are proportional gain, integral gain, and derivative gain respectively, $e(t_k)$ is a system error at the sample t_k, and Δ_t is a sampling time. In prior studies, the controller cannot effectively operate unless it must be set by specific tuning method such as conventional PID technique, LQR technique [1], trial-error technique [2], least mean square method (LMS) [3] and training technique [17]. In other words, the controllers insufficiently use for the nonlinear dynamic system unless the initial conditions are pre-set which are nearly the system stability. Furthermore, neural PID controller uses gradient algorithm to up-

Manuscript received on October 4, 2016 ; revised on November 24, 2016.
Final manuscript received on February 21, 2017.
[1,2] The authors are with the Department of Electronics Engineering, Faculty of Engineering, King Mongkuts Institute of Technology Ladkrabang, E-mail: adna@tni.ac.th and kkyuttan@kmitl.ac.th
[1] Corresponding author.

date the weights which has low speed of convergence, and is not able to tune weights for improving the error signal, the steady-state error, and the transient response of the system. As a result, we summarized that the control system which used NNPID controller can improve the performance in two objectives: 1) the initial value of the controller, and 2) the weight updating rule. Fortunately, as the growing of the computational intelligent algorithm in a last decade ago, many researchers have proposed the improvement the convergent speed using the filter model especially in the Kalman filter model which already improved the convergent speed of the neural network such as [9], [10], and [11], known as EKF algorithm. In the literature, the algorithm must form the status of the weights of the neural network in term of the dynamic model which can be given as follows [11]:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \omega_k \tag{2}$$

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{w}_k, \boldsymbol{e}_k, \boldsymbol{b}_k) + v_k \tag{3}$$

where $\boldsymbol{w}_k$ is a state vector of weights of the neural network, $\boldsymbol{h}(\cdot)$ is a measurement model function, $\boldsymbol{e}_k$ and bk denote the input and the bias of the network, respectively, ω_k and v_k are the process noise and the measurement noise, respectively. The variable k denotes the computational iterations. The first and second equations are known as the system equation and the measurement equation, respectively. Although, EKF algorithm helps the neural network to learn the system but itself is based on linear model thus the controller may cause a problem in the linearization process. Consequently, many researchers still develop the Kalman filter model, for example, Wan et al. [12], Xiaoyu Wang and Yong Huang [13], Huizhong Yang et al. [14] to support the highly nonlinear system [11]. One of the most famous algorithms is namely cubature Kalman filter (CKF) [15]. As a result, this algorithm is widely used in the computational intelligent learning approach of the real-world applications such as the neural network time series prediction applications [18, 19, 20], the neural network fitting applications [21], and especially in the neural network control system applications [22, 23]. Moreover, this algorithm is also proved in term of the system stability [24, 25].

As aforementioned, the algorithm has several advantages such as designing as nonlinear controller, providing derivative free method, and providing an efficient solution for high-dimensional problem. Therefore, we construct the new model of the NNPID controller using the CKF algorithm, gradient algorithm, and the multi-loop tuning, called Hybrid CKF-NNPID controller to improve the output response and the initial values. The rest of this paper is organized as follows. A proposed controller design is first described in section II. Then, in section III, the experiment setup and the comparison of the control technique performances between the proposed controller and the other controllers are discussed. Finally, the conclusions are given in section IV.

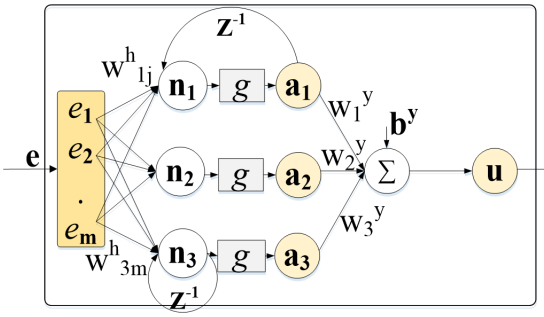

Fig.1: *Neural network controller based on PID structure.*

2. THE PROPOSED CONTROLLER DESIGN

The basic idea of the proposed controller is based on the fusion of the conventional PID controller with computational intelligent technique. The hybrid controller will make the controller increase the performances such as eliminate the restriction of the initial values setting, reduce the sensitivity of the weight update status and increase the speed of the convergence. The detail of the proposed controller will be described as follows.

2.1 The Proposed Controller Structure

The proposed controller structure comprises three layers, input layer, hidden layer, and output layer as shown in Fig. 1. In other words, the input layer receives the system error and then sends it out to the second layer which has three nodes. Each node in this layer behaves like the PID controller. Consequently, we define the first node as the integral node $\boldsymbol{a}_1$ that represents the integral gain $\mathbf{K}_\mathrm{I}$. In the second node, the $\boldsymbol{a}_2$ represents the propositional gain $\mathbf{K}_\mathrm{P}$, and the last node, the derivative node, $\boldsymbol{a}_3$ represents the derivative gain $\mathbf{K}_\mathrm{D}$ which can be expressed as follows:

$$a_{1,k} = g\left(\sum_{j=1}^{m} w_{j,k}^{h} e_{j,k} + a_{1,k-1}\Delta t\right) \tag{4}$$

$$a_{2,k} = g\left(\sum_{j=1}^{m} w_{2j,k}^{h} e_{j,k}\right) \tag{5}$$

$$a_{3,k} = g\left(\left(\sum_{j=1}^{m} w_{3j,k}^{h} e_{j,k} - \sum_{j=1}^{m} w_{3j,k-1}^{h} e_{j,k-1}\right)\frac{1}{\Delta t}\right) \tag{6}$$

where $\boldsymbol{m}$ is the number of the input nodes, $g(\cdot)$ is the nonlinear activation function that is set to tanh function. w_{ij}^{h} is the weight linked j^{th} input neuron with i^{th} hidden neuron, e_j is the j_{th} input of the neuron, and t is sampling time. The output layer is the summation of the hidden node, u_k can be expressed as

$$u_k = \sum_{i=1}^{3} w_{i,k}^{y} a_{i,k} + b^{y} \tag{7}$$

where w_i^y is the weight that links i^{th} hidden neuron with the output neuron, b^y is bias in the output layer set to zero.

2.2 The Proposed Weight Update Rule

The problem of the research is to iteratively filter the weights of the neural network PID controller which depend on the errors of the plant. Since the errors are considered as the nonlinear system signals with addition Gaussian noise, so the non-linear Kalman filter is appropriate tool for this, known as cubature Kalman filter (CKF). Therefore, in this paper, we propose the cubature Kalman filter (CKF) to update the weights of neural network PID controller, called Hybrid CKF-NNPID controller. The overall architecture of the Hybrid CKF-NNPID controller is shown in Fig. 2. It comprises two-loop operations, an external loop operation and an internal loop operation. The external loop operation is a neural network controller based on PID structure producing the control input for the plant which is subscripted by k. The other loop, internal loop, is superscripted by i that is denoted by dash line of the rectangle. This loop provides the weight update rule of the proposed controller which can improve initial values of the controller. In addition, we have redefined the system equation of the Kalman model to increase the controller performances but the measurement equation is still the same. Both of the equations are expressed as

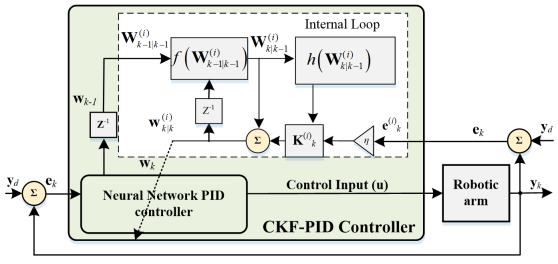

Fig.2: *The block diagram of the MIMO control system using a proposed controller.*

$$\boldsymbol{w}_{k+1} = \mathbf{f}(\boldsymbol{w}_k) + \omega_k \tag{8}$$

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{w}_k, \boldsymbol{e}_k, \boldsymbol{b}_k, u_k) + v_k \tag{9}$$

where $\boldsymbol{w}_k$ and $\boldsymbol{y}_k$ are a state vector and observation data, respectively, $\boldsymbol{e}_k$, $\boldsymbol{b}_k$ and u_k denote the input, the bias of the network and control input, respectively, ω_k and v_k are the process noise and the measurement noise, respectively. $\mathbf{h}(\cdot)$ is a measurement model function, and $\mathbf{f}(\cdot)$ is a process model function, known as an updated weights function, which is obtained by the gradient descent algorithm to update the internal weights. Each of the internal loop iterative operations, the updating predicted weights will be optimized by calculation the Kalman gain ($\mathbf{K}$). The algorithm will calculate until reaching the internal criteria. Then, the calculation will exit from the internal loop to the external loop and these weights are the new updated weights of the control input calculations. The external loop calculation will continually update the weights until gaining the criteria. The details of the proposed controller will be discussed as follows. The first process of the proposed controller is the formation of all of the parameters. From Fig. 1, lets $\boldsymbol{w}^y$ and $\boldsymbol{w}^h$ be a state of the weighted network in the output and the hidden layer, respectively that are expressed as

$$\boldsymbol{w}^h = \begin{bmatrix} w_{11}^h & \cdots & w_{1j}^h & b_1^h \\ w_{21}^h & \cdots & w_{2j}^h & b_2^h \\ w_{31}^h & \cdots & w_{3j}^h & b_3^h \end{bmatrix} \tag{10}$$

$$\boldsymbol{w}^y = \begin{bmatrix} w_1^y & w_2^y & w_3^y & b^y \end{bmatrix}^T \tag{11}$$

Therefore, the state vector of the Kalman model is given by the weighted network represented as

$$\boldsymbol{w}_k = \begin{bmatrix} w_{11,k}^h & \cdots & b_{3,k}^h & w_{1,k}^y & \cdots & w_{3,k}^y & b_k^y \end{bmatrix}^T \tag{12}$$

According to the basic of the dynamic neural network, the output function can be expressed as

$$u_k = \boldsymbol{w}_k^y g\left(\left[\boldsymbol{w}_k^h \boldsymbol{x}_k\right] + \mathbf{b}^h\right) + b^y \tag{13}$$

where $\mathbf{x}$ is the network input that will be discussed later. The $g(\cdot)$ is the activation function in the hidden layer which uses a tanh function. Where b^y is set to zero, and $\mathbf{b}^h$ is PID characteristic matrix that follows Eq. (4), Eq. (5), and Eq. (6), respectively, which is given by

$$\mathbf{b}^h = \left[a_{1,k-1}; 0; -\sum_{j=1}^{m} w_{3j,k-1}^h e_{j,k-1} \right] \tag{14}$$

The plant will give the observation values, also called $\mathbf{z}_k$. It will be compared with the desired value ($\mathbf{y}_d$) to calculate the error signals ($\mathbf{e}_k$) which is formed as

$$\mathbf{x}_k = \left[e_1 \ \cdots \ e_m\right]_k \tag{15}$$

where $\mathbf{x}_k$ is an inputs of the neural network. Then, weight update rule of the proposed controller shows in Fig. 2 which has the procedure as follows, First, at the beginning of the operations, all of the parameters including state vector ($\mathbf{w}$), Kalman gain ($\mathbf{K}$), error covariance ($\mathbf{P}$), and measurement noise covariance ($\mathbf{R}$) will be initiated by random to generate the control input (u) for the plant. Consequently, the external loop of the system will give the error signals ($\mathbf{e}_k$) which is the difference between current angle position and desired angle position. Next, if error is not zero, the old weights ($\mathbf{w}_{k-1|k-1}$) in the weight update rule begin to be optimized by the CKF algorithm at the internal loop as shown in Fig. 2. The procedure of CKF algorithm will be described into 2 steps as follows [15]:

Step 1: The model prediction

The predicted state ($\hat{\mathbf{w}}_{k|k-1}$) and the predicted error covariance ($\mathbf{P}_{k|k-1}$) are first given by

$$\hat{\mathbf{w}}_{k|k-1} = \frac{1}{2n}\sum_{l=1}^{2n} \boldsymbol{W}^{*}_{l,k|k-1} \tag{16}$$

$$\mathbf{P}_{k|k-1} = \tfrac{1}{2n}\sum_{l=1}^{2n}\left(\boldsymbol{W}^{*}_{l,k|k-1}\boldsymbol{W}^{*T}_{l,k|k-1}\right) - \hat{\mathbf{w}}_{k|k-1}\hat{\mathbf{w}}^{T}_{k|k-1} + \mathbf{Q}_{k-1} \tag{17}$$

where $\mathbf{Q}_k$ is the process noise error covariance matrix that assumes to be zero, and the propagated cubature points ($\mathbf{W}^{*}_{l,k|k-1}$) is given by the process model function using the back propagation neural network algorithm [16] which can be expressed as

$$\mathbf{W}^{*}_{l,k|k-1} = f\left(\mathbf{W}_{l,k-1|k-1}\right) = \mathbf{W}_{l,k-1|k-1} + \alpha\tfrac{\partial \mathbf{E}^{(i)}_{k-1}}{\partial \mathbf{W}_{l,k-1|k-1}} \tag{18}$$

where $\mathbf{E}^{(i)}_{k-1}$ is an error cost function of the internal loop at i iteration or is the instanteneous external error of time $k-1$, which will be discussed latter, α is a learning rate that is set to 0.001, $\mathbf{W}_{l,k-1|k-1}$ is a cubature points of model prediction that can be expressed as

$$\mathbf{W}_{l,k-1|k-1} = \mathbf{S}_{k-1|k-1}\zeta_l + \mathbf{w}_{l,k-1|k-1} \tag{19}$$

$$\zeta_l = \begin{cases} \sqrt{n}\mathbf{I}_l & l = 1,2,\ldots,n \\ -\sqrt{n}\mathbf{I}_{l-n} & l = n+1, n+2, \ldots, 2n \end{cases} \tag{20}$$

where n is a quatity of the state vector, and $\mathbf{I}$ is the identity matrix with l^{th} column vector, $\mathbf{w}_{l,k-1|k-1}$ is a old update weight which obtains in step 2, and the variable $\mathbf{S}_{k-1|k-1}$ is a square root of the last corresponding error covariance matrix ($\mathbf{P}_{k-1|k-1}$) expressed as

$$\mathbf{P}_{k-1|k-1} = \mathbf{S}_{k-1|k1}\mathbf{S}^{T}_{k-1|k1} \tag{21}$$

Step 2: The measurement prediction

To optimize the predicted state ($\hat{\mathbf{w}}_{k|k-1}$), we must determine the Kalman gain expressed as

$$\mathbf{K}_k = \mathbf{P}_{xy,k|k-1}\left(\mathbf{P}_{yy,k|k-1}\right)^{-1} \tag{22}$$

where innovation covariance matrix ($\mathbf{P}_{yy,k|k-1}$) and the cross-covariance matrix ($\mathbf{P}_{xy,k|k-1}$) of the state vector ($\mathbf{w}_k$) and measurement output ($\mathbf{y}_k$) are given by

$$\mathbf{P}_{yy,k|k-1} = \tfrac{1}{2n}\sum_{2n}^{1}\left[\mathbf{h}(\mathbf{w}_{l,k|k-1})\mathbf{h}(\mathbf{w}_{l,k|k-1})^{T}\right] - \hat{\mathbf{y}}_{k|k-1}\hat{\mathbf{y}}^{T}_{k|k-1} + \mathbf{R}_k \tag{23}$$

$$\mathbf{P}_{xy,k|k-1} = \tfrac{1}{2n}\sum_{2n}^{1}\left[\mathbf{w}_{l,k|k-1}\mathbf{h}(\mathbf{w}_{l,k|k-1})^{T}\right] - \hat{\mathbf{w}}_{k|k-1}\hat{\mathbf{y}}^{T}_{k|k-1} \tag{24}$$

where $\mathbf{h}(\cdot)$ is neural network PID controller function and $\mathbf{R}_k$ is the noise measurent covariance that is given by

$$\mathbf{R}_k = \mathbf{R}_{k-1} + \frac{1}{k}\left((\boldsymbol{y}_d - \boldsymbol{y}_k)(\boldsymbol{y}_d - \boldsymbol{y}_k)^{T} - \mathbf{R}_{k-1}\right) \tag{25}$$

the term of the cubature points of the measurement prediction ($\mathbf{w}_{l,k|k-1}$) and predicted output ($\hat{\mathbf{y}}_{k|k-1}$) are given by

$$\mathbf{w}_{l,k|k-1} = \mathbf{S}_{k|k-1}\zeta_l + \hat{\mathbf{w}}_{k|k-1} \tag{26}$$

$$\hat{\mathbf{y}}_{k|k-1} = \frac{1}{2n}\sum_{l=1}^{2n}\mathbf{h}\left(\mathbf{w}_{l,k|k-1}\right) \tag{27}$$

where $\mathbf{S}_{k|k-1}$ is a square root of the predicted error covariance matrix ($\mathbf{P}_{k|k-1}$) that is given by step *1*. Then, the new update of the weights and the updated corresponding error covariance in the internal loop calculation are expressed as

$$\mathbf{w}_{k|k} = \hat{\mathbf{w}}_{k|k-1} + \eta\mathbf{K}_k(\boldsymbol{y}_d - \boldsymbol{y}_{k-1}) \tag{28}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{P}_{yy,k|k}\mathbf{K}^{T}_k \tag{29}$$

where η is the Kalman scale rate that is set to 0.1. It is the optimal value in according to the section 2.3 that will be discussed later. The CKF algorithm will evaluate the level of the incremented weights, known as Kalman gain ($\mathbf{K}$). From our experiments, the Kalman gain gives a large scale so it must be multiplied by the Kalman scale rate (η) to normalize and to avoid the over fitting in the increment of the weight update. These modifications increase the controller performances such as reduce the sensitivity with noises and improve the transient response of the control system. Finally, the recursive calculation in the internal loop will continually update the weights of the network until reaching the setting criteria that is set to 20 epochs. Then, the operation

switches from the internal loop to the external loop. Namely, these weights will replace the old weights of the control input calculations. The external loop calculation will continually update the weight until the system give zero value of the system error in according to the equation (30) otherwise the operations will be back to step 2 for the new update with the new system error.

$$\mathbf{E}_k = \frac{1}{2}\sum_{j=1}^{m}(\boldsymbol{y}_{d,j} - \boldsymbol{y}_{k,j})^2 \tag{30}$$

where $\mathbf{E}_k$ is an error cost function of the external loop at time k that is a sum sqaure function of the different value between desired angles position and current angles position.

2.3 The Control System Stability Analysis

To guarantee the control system stability, we apply the Lyapunovs theory to the robotic arm controller. In fact, the Lyapunovs function is only the method for the consideration of the nonlinear control system stability of general characteristic system. In other words, the control system that corresponds to the Lyapunovs function is a defined as a stable. In order to achieve the requirement, Kalman scale rate (η) have to be defined as following steps; firstly, the Lyapunovs function $\mathbf{V}_k$ is defined as

$$\begin{aligned}\mathbf{V}_k &= \frac{1}{2}\sum_{j=1}^{m}(\boldsymbol{y}_{d,j} - \boldsymbol{y}_{j,k})^2 \\ &= \frac{1}{2}\sum_{j=1}^{m} e_{j,k}^2\end{aligned} \tag{31}$$

next, the difference of the Lyapunovs function is given by substitute the equation (31) to below equations

$$\begin{aligned}\Delta\mathbf{V}_k &= \mathbf{V}_{k+1} - \mathbf{V}_k \\ &= \frac{1}{2}\sum_{j}^{m} e_{j,k+1}^2 - \frac{1}{2}\sum_{j}^{m} e_{j,k}^2 \\ &= \frac{1}{2}\sum_{j}^{m}[(e_{j,k+1} + e_{j,k})(e_{j,k+1} - e_{j,k})] \\ &= \frac{1}{2}\sum_{j}^{m}[(e_{j,k+1} - e_{j,k} + 2e_{j,k})(\Delta e_{j,k})] \\ &= \frac{1}{2}\sum_{j}^{m}[(\Delta e_{j,k} + 2e_{j,k})(\Delta e_{j,k})] \\ &= \frac{1}{2}\sum_{j}^{m}\left[(2\Delta e_{j,k}e_{j,k}) + (\Delta e_{j,k})^2\right]\end{aligned} \tag{32}$$

In order to find the difference of the system error (Δe_k), the gradient method is utilized which can be expressed as

$$\begin{aligned}\Delta e_k &= \frac{\partial E_k}{\partial k} = \frac{\partial E_k}{\partial \mathbf{W}_k}\frac{\partial \mathbf{W}_k}{\partial k} \\ &= \frac{\partial(y_{d,k} - y_k)}{\partial \mathbf{W}_k}\Delta\mathbf{W}_k\end{aligned} \tag{33}$$

substitute the $\Delta\mathbf{W}_k$ from the equation (28) that is equal to $\eta\mathbf{K}_k\Delta e_k$ into the equation (33), then

$$\Delta e_k = \frac{\partial y_k}{\partial \mathbf{W}_k}\eta_k\mathbf{K}_k e_k \tag{34}$$

substitute the Δe_k to the equation (32). Then, the difference of the Lyapunovs function is defined as

$$\Delta V_k = \sum_{j=1}^{m}\left[\left(-\tfrac{\partial y_k}{\partial \mathbf{W}_k}\eta_k\mathbf{K}_k e_{j,k}^2\right) + \tfrac{1}{2}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}\eta_k\mathbf{K}_k e_{j,k}\right)^2\right] \tag{35}$$

According to Lyapunovfs stability theorem, the system is stable only if $\Delta V \leq 0$ then, η is defined as the following criteria:

case 1: if $\sum_{j=1}^{m}\left[\left(\frac{\partial y_k}{\partial \mathbf{W}_k}\mathbf{K}_k e_{j,k}^2\right)\right] < 0$, then

$$\sum_{j=1}^{m}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}\eta_k\mathbf{K}_k e_{j,k}^2\right) + \sum_{j=1}^{m}\tfrac{1}{2}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}\mathbf{K}_k e_{j,k}\right)^2 \leq 0$$

$$\eta_k\mathbf{K}_k\sum_{j=1}^{m}\tfrac{1}{2}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}\right)^2 \leq -\sum_{j=1}^{m}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}^2\right)$$

$$\eta_k \leq -2\frac{\sum_{j=1}^{m}\left(\frac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}^2\right)}{\mathbf{K}_k\sum_{j=1}^{m}\left(\frac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}\right)^2} \tag{36}$$

case 2: if $\sum_{j=1}^{m}\left[\left(\frac{\partial y_k}{\partial \mathbf{W}_k}\mathbf{K}_k e_{j,k}^2\right)\right] \geq 0$, then

$$\sum_{j=1}^{m}\left(-\tfrac{\partial y_k}{\partial \mathbf{W}_k}\eta_k\mathbf{K}_k e_{j,k}^2\right) + \sum_{j=1}^{m}\tfrac{1}{2}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}\mathbf{K}_k e_{j,k}\right)^2 \leq 0$$

$$\eta_k\mathbf{K}_k\sum_{j=1}^{m}\tfrac{1}{2}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}\right)^2 \leq \sum_{j=1}^{m}\left(\tfrac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}^2\right)$$

$$0 \leq \eta_k \leq 2\frac{\sum_{j=1}^{m}\left(\frac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}^2\right)}{\mathbf{K}_k\sum_{j=1}^{m}\left(\frac{\partial y_k}{\partial \mathbf{W}_k}e_{j,k}\right)^2} \tag{37}$$

where matrix $\mathbf{K}_k$ is Kalman gain that is given in the previous section.

3. EXPERIMENTS AND RESULTS

3.1 Experiment Setup

The aim of this experiment is to reduce an angle error of the arm movements with the best transient response conditions. The experiment is modeled by

the Matlab program as shown by a block diagram in Fig. 3. The demonstrated block diagram has two major components, which are the controller and the plant. In order to test the performance of the proposed controller, we create other three controllers: 1) ENNPID controller, 2) NNPID controller, controller and 3) conventional PID controller to compare with the proposed controller. All of the controllers except the conventional PID controller are created by the Matlab/Simulink using embedded Matlab function. Inside the block, it contains the listing codes based-on the C language program in according to the gradient algorithm [6] and extended Kalman filter algorithm [17], respectively. In the case of the PID controller, we use the standard libraries in the Matlab. Generally, tuning method of the PID gains the system parameters must be known, so we adds the parameters of the DC motor along with the armature inductance, stall torque, no-load speed, dc supply voltage, and the no-load current which are set with the values of 0.12 H, 0.402 Nm, 316 rad/s, 6v, and 8 mA, respectively. Consequently, we obtain K_P, K_I, and K_D, which are 25.0, 1.5, and -1.0, respectively. These values are also defined as the initial values of the weighted networks of the NNPID controller. In the case of the ENNPID controller, the initial values is set by trained values without the use of the system parameters but the controller still need the pre-training data before using it. In contrast, the initial values of the proposed controller is set by random within some interval without the use of the system parameters and pre-training data. Consequently, the proposed controller is easily applied to the control system in various conditions.

The 4-joint robotic arm structure is also created by using the Matlab/Simulinks library. The detail of the robotic arm structure is shown by the graphical view of all the joints that displays in Fig. 4. In the experiment, since system error is the error of the joint angle of the robotic arm, we put a controller for each of the joints. An angle error of that joint is the input of the controller. The sampling time of the system is equal to 0.01 second. Furthermore, the constraints of the angle movements are also defined to protect the non-sense movements of the robotic arm and we also create the multi-step input signal as the reference angles to simulate the 4-joint robotic arm movements. Then, the results of the experiments of all types of the controllers are compared the performances with each other.

3.2 Experiment Results

To verify the proposed controller, we apply the multistep input to the 4-joint robotic arm control system with various types of the controllers including CKF-NNPID controller, ENNPID controller, NNPID controller and conventional PID controller. In Fig. 5, the 4-joint robotic arm system along with (a) turntable joint, (b) bicep joint, (c) forearm joint, and

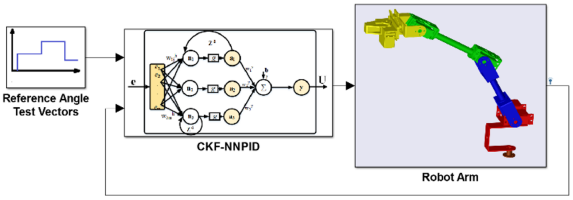

Fig.3: *The block diagram of simulation for the robotic arm control system using the proposed controller.*

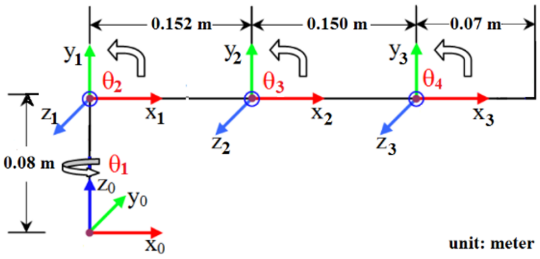

Fig.4: *The coordinate frames of the robotic arm.*

(d) wrist joint under the control of the proposed controller gives the best performance in terms of the maximum overshoot, steady state error, and rise time that are highly require in the control system evaluation because the proposed controller can regenerate the gains to create the control input to compensate the system if the faults have taken place. It clearly see that the CKF-NNPID controller is able to search the values of the gains, initial values, while the classical controller such as the conventional PID control requires the new PID gains (KI, KP, and KD) if the conditions have been changed. Moreover, simulation results show that the algorithm still gives the best result even though the robotic arm is under the loads comparing with the conventional PID control or the other neural PID controller as shown in Fig. 6.

Since we know that the controllers based on the neural network have some disadvantages such as over fitting, large data training requirements, slow convergent rate, especially in the initial value sensitivity. In order to eliminate the initial value sensitivity problem, we initiate the values for the weights of the proposed controller using the random method with 10 different values to control the robotic arm. As a result, it obviously see that the proposed controller applied with 10 different values can track the errors of each joint angle and eventually converges to zero as shown in Fig. 7, Fig. 8, Fig. 9, and Fig.10 for turntable joint, bicep joint, forearm joint, and wrist joint, respectively.

4. CONCLUSIONS

This paper investigates the online tuning neural network PID controller to a nonlinear robotic arm to

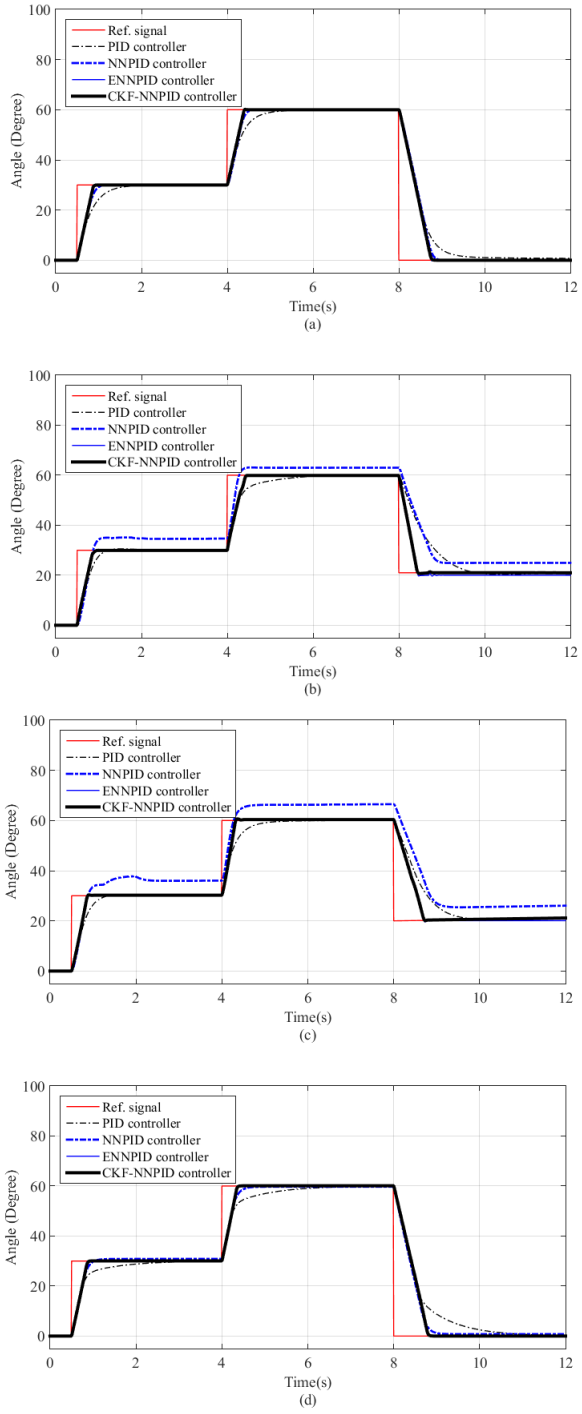

Fig.5: *The angle response comparison of the MIMO control system between the proposed controller and other controllers without load, (a) Turntable joint, (b) Bicep joint, (c) Forearm joint, and (d) Wrist joint.*

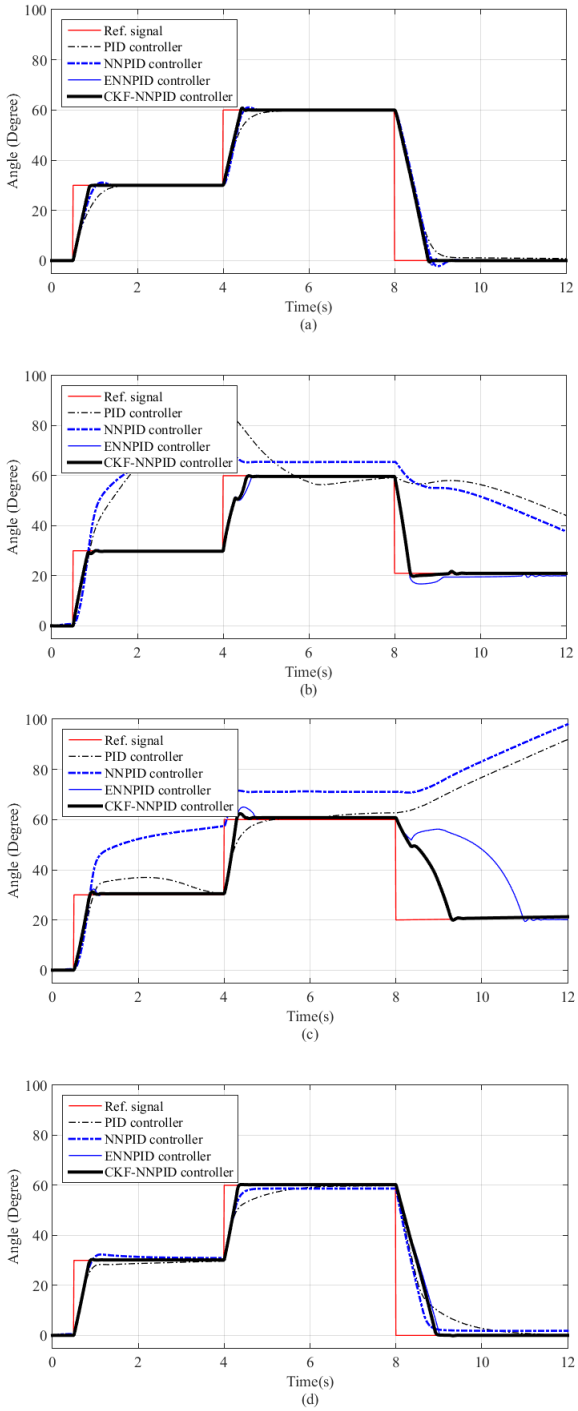

Fig.6: *The angle response comparison of the MIMO control system between the proposed controller and other controllers with load, (a) Turntable joint, (b) Bicep joint, (c) Forearm joint, and (d) Wrist joint.*

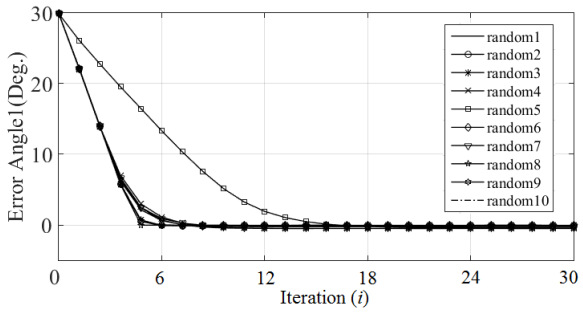

Fig.7: *The error trajectories of the turntable angle of the proposed controller with 10 of the initial values.*

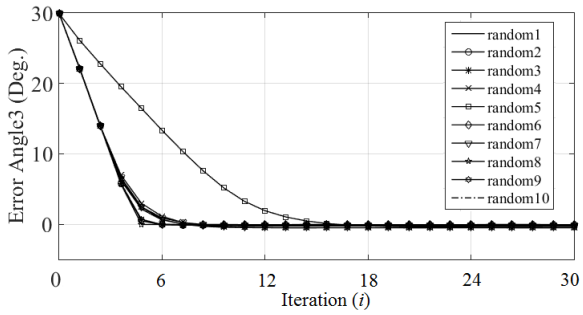

Fig.8: *The error trajectories of the bicep joint of the proposed controller with 10 of the initial values.*

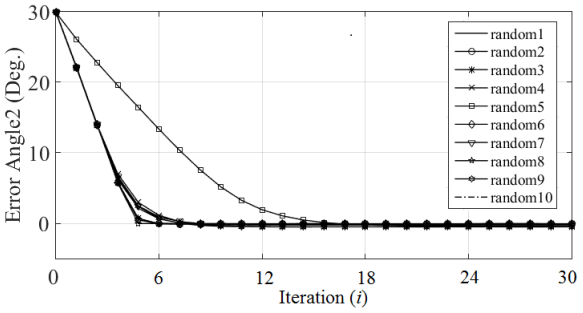

Fig.9: *The error trajectories of the forearm of the proposed controller with 10 of the initial values.*

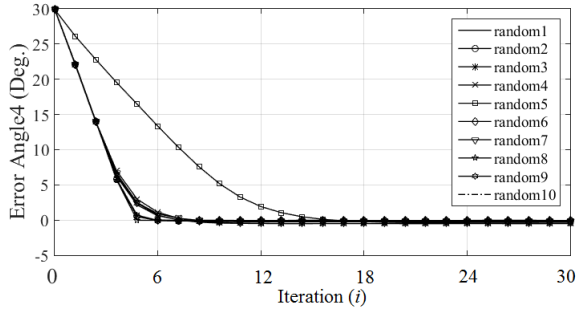

Fig.10: *The error trajectories of the wrist of the proposed controller with 10 of the initial values.*

improve its joint angle position output performance. We use multi-loop operations of the hybrid of the gradient descent algorithm with cubature Kalman filter algorithm. The controller will predict the weights in each internal iteration and then optimize them by cubature Kalman filter (CKF) algorithm until reaching the internal optimal weights, after that these weights will be used to create the compensated voltage control input of the robotic arm system, called Hybrid CKF-NNPID controller. In order to evaluate the controller performances, we apply the proposed controller to 4-joint robotic arm on the MATLAB simulation. From our experimental results, it shows that the proposed controller is a superior controller comparing with the others controller such as conventional PID controller, NNPID controller, and ENNPID controller because the proposed controller potentially tracks the error system even though the system is under the loading criteria. It also improves the convergent speed and stability sustainable. Furthermore, the proposed controller has several advantages for example, it can serve for online tuning without the need of the system parameters to tune the initial gain values while the prior controller must have the system parameters to tune the PID gains, or by trialand- error or by training sampling data.

References

[1] S. Cong and Y. Liang, "PID-Like Neural Network Nonlinear Adaptive Control for Uncertain Multivariable Motion Control Systems," in *IEEE Transactions on Industrial Electronics*, vol. 56, no.10, pp. 3872-3879, Oct. 2009.

[2] H. Huy Anh, "Online tuning gain scheduling MIMO neural PID control of the 2-axes pneumatic artificial muscle (PAM) robot arm," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6547-6560, 2010.

[3] W. Yu and J. Rosen, "Neural PID Control of Robot Manipulators With Application to an Upper Limb Exoskeleton," in *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 673-684, April 2013.

[4] V. Kumar, P. Gaur and A. Mittal, "ANN based self tuned PID like adaptive controller design for high performance PMSM position control," *Expert Systems with Applications*, vol. 41, no. 17, pp. 7995-8002, 2014.

[5] F. Rossomando and C. Soria, "Design and Implementation of Adaptive Neural PID for Non Linear Dynamics in Mobile Robots," in *IEEE Latin America Transactions*, vol. 13, no. 4, pp. 913-918, April 2015.

[6] J. L. Meza, R. Soto and J. Arriaga, "An Optimal Fuzzy Self-Tuning PID Controller for Robot Manipulators via Genetic Algorithm," *Artificial Intelligence, 2009. MICAI 2009. Eighth Mexican*

International Conference on, Guanajuato, 2009, pp. 21-26.

[7] J. L. Meza, V. Santibanez, R. Soto and M. A. Llama, "Fuzzy Self-Tuning PID Semiglobal Regulator for Robot Manipulators," in *IEEE Transactions on Industrial Electronics*, vol. 59, no. 6, pp. 2709-2717, June 2012.

[8] J. Armendariz, V. Parra-Vega, R. Garca-Rodrguez and S. Rosales, "Neuro-fuzzy self-tuning of PID control for semiglobal exponential tracking of robot arms," *Applied Soft Computing*, vol. 25, pp. 139-148, 2014.

[9] Y. Iiguni, H. Sakai and H. Tokumaru, "A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter," in *IEEE Transactions on Signal Processing*, vol. 40, no. 4, pp. 959-966, Apr 1992.

[10] I. Rivals and L. Personnaz, "A recursive algorithm based on the extended Kalman filter for the training of feedforward neural models," *Neurocomputing*, vol. 20, no. 1-3, pp. 279-294, 1998.

[11] S. Haykin, *Kalman filtering and neural networks*. New York: Wiley, 2001.

[12] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, Lake Louise, Alta., 2000, pp. 153-158.

[13] X. Wang and Y. Huang, "Convergence Study in Extended Kalman Filter-Based Training of Recurrent Neural Networks," in *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 588-600, April 2011.

[14] H. Yang, J. Li and F. Ding, "A neural network learning algorithm of chemical process modeling based on the extended Kalman filter," *Neurocomputing*, vol. 70, no. 4-6, pp. 625-632, 2007.

[15] I. Arasaratnam and S. Haykin, "Cubature Kalman Filters," in *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1254-1269, June 2009.

[16] D. Rumelhart, G. Hinton and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533-536, 1986.

[17] A. Sento and Y. Kitjaidure, "Neural network controller based on PID using an extended Kalman filter algorithm for multi-variable nonlinear control system," *2016 Eighth International Conference on Advanced Computational Intelligence (ICACI)*, Chiang Mai, 2016, pp. 302-309.

[18] C. Sheng, J. Zhao, Y. Liu and W. Wang, "Prediction for noisy nonlinear time series by echo state network based on dual estimation," *Neurocomputing*, vol. 82, pp. 186-195, 2012.

[19] B. Safarinejadian, M. A. Tajeddini and A. Ramezani, "Predict time series using extended, unscented, and cubature Kalman filters based on feed-forward neural network algorithm," *Control, Instrumentation, and Automation (ICCIA), 2013 3rd International Conference on*, Tehran, 2013, pp. 159-164.

[20] A. Mitchell, "Online Courses and Online Teaching Strategies in Higher Education," *Creative Education*, vol. 05, no. 23, pp. 2017-2019, 2014.

[21] I. Arasaratnam and S. Haykin, "Nonlinear Bayesian Filters for Training Recurrent Neural Networks," *MICAI 2008: Advances in Artificial Intelligence: 7th Mexican International Conference on Artificial Intelligence*, pp. 12-33, 2008.

[22] A. Chernodub, "Local control gradients criterion for selection of neuroemulators for model reference adaptive neurocontrol," *Optical Memory and Neural Networks*, vol. 21, no. 2, pp. 126-131, 2012.

[23] F. Yang, W. Sun, G. Lin and W. Zhang, "Prediction of Military Vehicles Drawbar Pull Based on an Improved Relevance Vector Machine and Real Vehicle Tests," *Sensors*, vol. 16, no. 3, p. 351, 2016.

[24] J. Zarei and E. Shokri, "Convergence analysis of nonlinear filtering based on cubature Kalman filter," in *IET Science, Measurement & Technology*, vol. 9, no. 3, pp. 294-305, 5 2015.

[25] S. Wang, J. Feng and C. Tse, "Novel cubature Kalman filtering for systems involving nonlinear states and linear measurements," *AEU - International Journal of Electronics and Communications*, vol. 69, no. 1, pp. 314-320, 2015.

Adna Sento received Bachelor of Engineering from Thammasat University, Pathumthani Thailand, and Master of Engineering from King Mongkut's Institute of Technology Ladkrabang, Bangkok Thailand. Currently, He is studying a Ph.D. degree in Electrical Engineering at King Mongkut's Institute of Technology Ladkrabang, and He is lecture at department of the Computer Engineering, Thai-Nichi Institute of Technology.

Yuttana Kitjaidure received Bachelor of Engineering and Master of Engineering from King Mongkut's Institute of Technology Ladkrabang, Bangkok Thailand, and Ph.D. in Electronic and Electrical Engineering, Imperial College, University of London, England. Currently, He is a lecture at department of the Electronics Engineering, King Mongkut's Institute of Technology Ladkrabang.