Received: August 1, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Neural Network PID-Like Controller Using a Hybrid of Online Actor-Critic Reinforcement Algorithm with the Square Root Cubature Kalman Filter

Adna Sento [1]* **Yuttana Kitjaidure [1]***

[1] *Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand*
* Corresponding author's Email: adna@tni.ac.th

Abstract: This paper presents a new model of the Neural Network PID-Like controller using an Actor-Critic reinforcement algorithm, called the Neural Network PID-Like controller using an Actor-Critic reinforcement algorithm (NNPID-AC). The proposed NNPID-AC controller is designed to develop the performances and the speed of calculation under the iterative learning algorithm. In the learning algorithm, the critic algorithm receives the reward value and control input to criticize the current state using the action-state value function approximation. Furthermore, instead of applying every available action to predict the local successor state, the algorithm only uses one-step estimation using the fifth degree spherical-radial cubature rule algorithm. To evaluate the proposed NNPID-AC controller, the robot arm MATLAB simulations have been implemented and provide the control system with the load and noise to prove the robustness and fault tolerance, respectively. From the results, the robot arm control system simulation under the control of the proposed NNPID-AC controller can potentially track the error and gives the best responses compared with the other conventional controller either with or without the load and the noise disturbance.

Keywords: Nonlinear dynamic control system, Actor-critic reinforcement learning algorithm, Neural network controller, Intelligent system.

1. Introduction

Generally, control system performances along with accuracy, precision, and responses of the dynamic control system, can be strongly affected by factors such as uncertain variables of models, noises of the measurements, an online control system, loading conditions and a non-linearity. In the past, the well-known controller such a PID controller is widely used on improving the control system applications to deal with these factors. Many studies of the controller that has been carried out need some algorithms to acquire the best performances. For example, an analytical tuning method requires an accurate plant model and an objective model to obtain the PID gain [1, 2]. Ziegler–Nichols' (Z-N) straightforward tuning technique is necessary to use an empirical test [3]. A frequency response method is suitable for the off-line tuning technique [4]. An optimization method requires the off-line numerical optimization method for the multi-objective [5]. Especially, an adaptive tuning method utilizes the computational intelligent technique has some limitations, including data preparation, over fitting values, time-consuming, etc [6-9].

Recently, the controller algorithms have been published to increase the performances such as a fuzzy controller [10], a Lyapunov gain PID (LGPID) algorithm [11], a neural network controller [12, 13], and a fractional order PID (FOPID) algorithm [14]. One of the successful controllers is the neural network controller based on PID architecture. This controller uses a combination of the computational intelligent algorithm and the PID controller, known as the neural network PID-Like controller (also called the NNPID controller). The performance of the NNPID controller depends on the weight updating rule. Due to the advantages of the learning algorithm such as robustness, an adaptation, and a multi-input multi-output system of the NNPID controller, they have been extensively applied to the weight updating rule for the dynamic control system. For examples, Kalman learning algorithm is used in the learning algorithm [15-17]. However, they still have the problem in the linearization of the highly nonlinear system. To solve this problem, the Cubature Kalman Filter (CKF) algorithm has been published [18] to deal with the linearization problem. Consequently, the results of the control system under the algorithm give good performances. Since the CKF algorithm is designed for the nonlinear system, it has been widely used in control applications [19-21].

More recent learning approaches have been published in control system applications, known as the actor-critic reinforcement algorithm [22-24]. In these papers, they use the neural network to construct the system model. However, the model is based on constant weight values. Then the main problem is left for the real-time nonlinear control system in terms of divergence of some initial values, and system performances. Therefore, in this paper, we construct the neural network controller using the new design of the iterative learning algorithm, namely a hybrid of online actor-critic reinforcement learning algorithm with the square root cubature Kalman filter, called the NNPID-AC controller. The proposed NNPID-AC controller uses the neural network to create the neural network actor controller algorithm and the critic algorithm. We use the action-state value approximation that is generated by the critic algorithm to reinforce the control input. The control input is developed by the weights of the neural network actor controller. The contributions of this paper are the guided target of the amplitude of control inputs using the action-state value function, which uses the fifth degree of the square root spherical-radial cubature rule algorithm without using the system parameters, and without applying for every available action to evaluate the local optimal state.

To evaluate the proposed NNPID-AC controller capabilities, the robot arm control system simulation using the MATLAB program is implemented. In the experiments, the NNPID-AC controller and other controllers, including the classical PID controller, the NNPID gradient-based controller [15] and the Hybrid-CKF controller [19] have been created. Furthermore, load and noise disturbance are added MATLAB simulations to test the robustness and fault tolerance of the system. The results of the proposed NNPID-AC controller from the robot arm simulations exhibit the best responses with various situations.

In the rest of this paper, the problem formulation is described in section 2. Next, the proposed controller design, including the model and the reinforcement learning algorithm, is presented in section 3. After that, experiment setup and their results using the MATLAB program are demonstrated in section 4 and section 5, respectively. These sections also provide the performance comparison between the proposed controller, and the other designs. Finally, the conclusion is given in section 6.

2. The problem formulation

Let us consider the plant moving by the controller under the closed-loop control system as shown in Fig. 1. In this study, the iterative learning algorithm is used to update the parameters. To create the iterative learning algorithm, the dynamic system model is first discussed. Since the proposed controller is based on the neural network, the weight update is significant to the system output performances. Then, the weight and the control input of the controller are defined as the state and the output of the dynamic system. And the relation between the weights and the output, also called the action, is a stochastic nonlinear system because of the activation function. Therefore, the dynamic system model is defined as

$$\mathbf{w}_{k+1} \approx p(\mathbf{w}_{k+1} \mid \mathbf{w}_k) \tag{1}$$

$$u_k = h\left(\mathbf{w}_k\right) \tag{2}$$

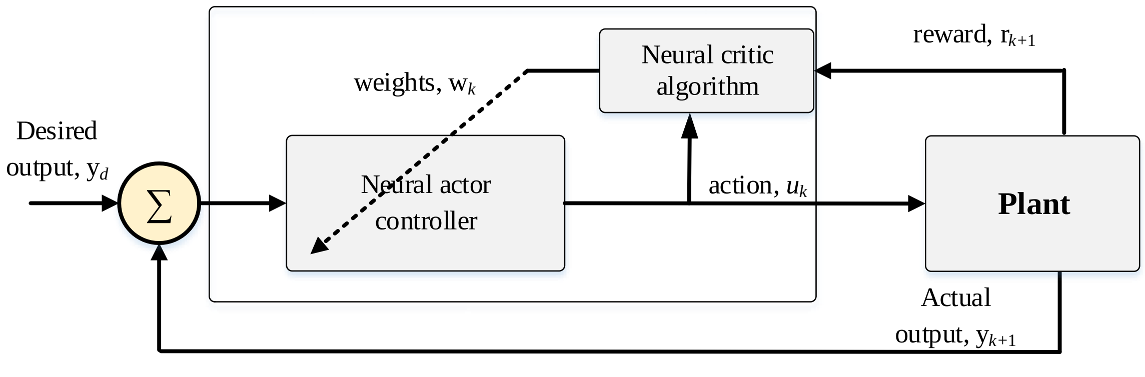

Figure. 1 The nonlinear dynamic control system diagram using the proposed NNPID-AC controller algorithm

where $\mathbf{w}_k$ is the state at the iteration kth, u_k is the control input of the system, $p(\mathbf{w}_{k+1}|\mathbf{w}_k)$ is the transition weight conditional function, and $h(\mathbf{w}_k)$ is the control input function that is the neural network PID-Like controller function. It will be described later in the topic of the neural network actor controller.

Regarding the state, the relation between the weights and the actions is nonlinear because of the activation function. Then, the next state and final state or desired target state are unknown. To find the next state applying to the system, the total cost must be minimized. According to the Bellman optimality principle [26], let $V(\mathbf{w},u)$ be a total cost function of the control system problem, also known as the value function approximation, given by

$$V_{k+1}(\mathbf{w},u) = \frac{1}{2}\left(y_d - y_k\right)^2 + E\left[V_k\left(\hat{\mathbf{w}}\right)\right] \tag{3}$$

where y_d and y_k denote the desired target value and the actual value of the control system at the iteration kth, respectively. $E[V_k(\hat{\mathbf{w}})]$ is the expected function of the value function of the optimal next state, $\hat{\mathbf{w}}$, called the local successor state. Considering the second term of the value function approximation, the large space of the control input is used to search for the local successor state. To reduce the high searching space, optimal action of the second term of the value function approximation is estimated. Therefore, the goal of this work is to search for the local successor state and the final successor state using a hybrid of online actor-critic reinforcement learning algorithm with the square root cubature Kalman filter until reaching the total cost function condition.

3. The proposed controller design

From Fig. 1, the control system diagram using the proposed NNPID-AC controller algorithm is divided into two operations; the neural network actor controller operation and the critic algorithm operation, called the neural actor and the critic algorithm, respectively. The control system architecture is designed by the neural network under the iterative learning algorithm, which begins generating control input using the neural actor to supply the plant. At each iteration, the actions will be evaluated by the critic algorithm using the action-state value function approximation that grades the state estimation for the best performances. In other words, the critic algorithm is designed to predict the unknown local successor state instead of searching the optimal action from all actions. The neural actor and the critic algorithm operation will be discussed in sequels.

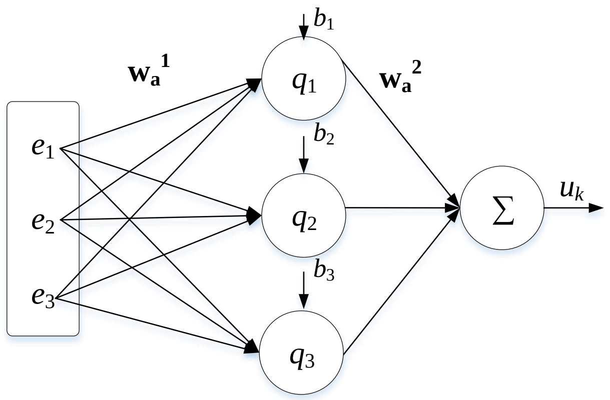

Figure. 2 The neural actor controller

3.1 The neural actor

To search for the optimal state, the neural actor will recursively calculate the weight to compensate the dynamic control system. The neural actor has been extended from the PID-Like controller according to [19], which consists of three-layer, namely the input layer, hidden layer, and output layer as shown in Fig. 2. The output of the network is given by

$$u_k = \mathbf{w}_{a,k}^2\left(\frac{1}{1+\exp^{-\left(\mathbf{w}_{a,k}^1 e_k + \mathbf{b}_k\right)}}\right), \mathbf{b}_k = \left[0, q_{1,k-1}, -q_{3,k-1}\right] \tag{4}$$

where e_k is the system error, which is the different value between the current value and the desired target at iteration kth. $q_{i,k}$ is the ith neuron at time k, called the bias value. $\mathbf{w}_a^1$ and $\mathbf{w}_a^2$ are the weights of the hidden layer and output layer, respectively. This network will update the optimal weights according to the value function approximation using the critic algorithm.

3.2 The critic algorithm

To update the weight of the neural actor, the critic algorithm has been used in the action-state value function approximation. The control input from the neural actor and the reward from the plant are the input of the critic algorithm for the action-state value function evaluation. The local successor state, also known as the weights of the controller, is first evaluated using the square root cubature Kalman filter instead of the considering every action available to reduce a searching space and a time consumption. According to the linear optimal estimation [25], we first assume that the relation of the state and the actor is Gaussian given by

$$p(\mathbf{w}_k,u_k)=N\left(\begin{bmatrix}\mathbf{w}_k\\ u_k\end{bmatrix};\begin{bmatrix}\hat{\mathbf{w}}_k\\ \hat{u}_k\end{bmatrix},\begin{bmatrix}\mathbf{\Sigma}_{ww,k|k} & \mathbf{\Sigma}_{wu,k|k}\\ \mathbf{\Sigma}_{uw,k|k} & \mathbf{\Sigma}_{uu,k|k}\end{bmatrix}\right) \quad (5)$$

where $N(\cdot)$ is the Gaussian distribution function with the local successor state ($\hat{\mathbf{w}}_k$) and error covariances ($\mathbf{\Sigma}_{k|k}$). Then, the conditional density of the Gaussian distribution can be estimated by

$$p(\mathbf{w}_k\,|\,u_k)\approx N(\mathbf{w}_k;\hat{\mathbf{w}}_k,\mathbf{\Sigma}_{k|k}) \quad (6)$$

where the local successor state is provided by

$$\hat{\mathbf{w}}_k=\hat{\mathbf{w}}_{k|k-1}+K\left(cost_k\right) \quad (7)$$

where $cost_k$ is the total cost function of the control system problem that will be replaced with the action-state value function approximation as described in the previous section. K is the Kalman gain given by

$$K=\mathbf{\Sigma}_{wu,k|k-1}\mathbf{\Sigma}^{-1}_{uu,k|k-1} \quad (8)$$

In case of the error joint covariance, it can be expressed as

$$\mathbf{\Sigma}_{k|k}=\mathbf{\Sigma}^{-1}_{ww,k|k-1}-\mathbf{\Sigma}_{wu,k|k-1}\mathbf{\Sigma}^{-1}_{uu,k|k-1}\mathbf{\Sigma}_{uw,k|k-1} \quad (9)$$

where $\mathbf{\Sigma}_{ww,k|k-1}$, and $\Sigma_{uu,k|k-1}$ are the predicted joint covariance and the innovation covariance matrix, respectively, which can be expressed as

$$\begin{aligned}\mathbf{\Sigma}_{ww,k|k-1}&=\int_{\Re}\mathbf{w}_{k-1}\mathbf{w}^T_{k-1}N(\mathbf{w}_{k-1};\hat{\mathbf{w}}_{k-1},\mathbf{\Sigma}_{k-1|k-1})d\mathbf{w}_{k-1}\\ &\quad-\mathbf{w}_{k|k-1}\mathbf{w}^T_{k|k-1}+\mathbf{Q}_{k-1}\end{aligned} \quad (10)$$

$$\begin{aligned}\mathbf{\Sigma}_{uu,k|k-1}&=\int_{\Re}h(\mathbf{w}_k)h(\mathbf{w}_k)^T N(\mathbf{w}_k;\hat{\mathbf{w}}_{k|k-1},\mathbf{\Sigma}_{k|k-1})d\mathbf{w}_k\\ &\quad-u_{k|k-1}u^T_{k|k-1}+\mathbf{R}_k\end{aligned} \quad (11)$$

Both of $\mathbf{\Sigma}_{wu,k|k-1}$ and $\Sigma_{uw,k|k-1}$ are the cross-covariance of the state and measurement given by

$$\Sigma_{wu,k|k-1}=\int_{\Re}\mathbf{w}_k h(\mathbf{w}_k)N(\mathbf{w}_k;\hat{\mathbf{w}}_{k|k-1},\Sigma_{k|k-1})d\mathbf{w}_k-\mathbf{w}_{k|k-1}u^T_{k|k-1} \quad (12)$$

where $u_{k|k-1}$ and $\hat{\mathbf{w}}_{k|k-1}$ are the predicted likely-hood function and the predicted state function, respectively, given by

$$\hat{\mathbf{w}}_{k|k-1}=\int_{\Re}\mathbf{w}_{k-1}N(\mathbf{w}_{k-1};\hat{\mathbf{w}}_{k-1},\mathbf{\Sigma}_{ww,k-1|k-1})d\mathbf{w}_{k-1} \quad (13)$$

$$u_{k|k-1}=\int_{\Re}h\left(\mathbf{w}_k\right)N(\mathbf{w}_k;\hat{\mathbf{w}}_{k|k-1},\mathbf{\Sigma}_{ww,k|k-1})d\mathbf{w}_k \quad (14)$$

where $h(\mathbf{w}_k)$, $\mathbf{Q}_{k-1}$ and $\mathbf{R}_k$ are the control law model that is the neural actor, the last error covariance and error output covariance, respectively. $\mathbf{w}_{k-1}$ is the last updated state. Finally, the term of $\mathbf{Q}_{k-1}$ is set to zero and $\mathbf{R}_k$ is given by

$$\mathbf{R}_k=\mathbf{R}_{k-1}+\frac{1}{k}\left(cost_k\left(cost_k\right)^T-\mathbf{R}_{k-1}\right) \quad (15)$$

where $\mathbf{R}_{k-1}$ is the last error covariance. Eq. (7) is the posterior distribution of the state updating function. It will be used in the closed-loop control system.

Regarding to the Gaussian integral function with the nonlinear function of $\int h(\mathbf{w})\mathrm{N}(\mathbf{w};\hat{\mathbf{w}},\sum)d\mathbf{w}$, it can be approximated by using the spherical-radial algorithm [18] that can be expressed as

$$\begin{aligned}I_N\left(h\right)&=\frac{1}{\sqrt{\pi^n}}\int_{\Re}h\left(\sqrt{2\mathbf{\Sigma}_{k|k-1}}\mathbf{w}_{k-1}+\hat{\mathbf{w}}_{k|k-1}\right)\\ &\quad\times\exp(-\mathbf{w}_{k-1}\mathbf{w}^T_{k-1})d\mathbf{w}_{k-1},\end{aligned} \quad (16)$$

where $\exp(-\mathbf{w}_k\mathbf{w}^{\mathrm{T}}_k)$ is the weighting function. To solve the above Gaussian weighting integral functions, the spherical-radial algorithm has been applied to transform them into the spherical-radial integral functions. Let r and s be $r=\mathbf{w}_k\mathbf{w}^{\mathrm{T}}_k$ and $s=\mathbf{w}_k/r$. Then, Eq. (16) can be approximated by

$$\begin{aligned}I_{\mathrm{N}}\left(h\right)&=\int_0^\infty\int_{\Re}h\left(r\mathbf{s}_k\right)r^{n-1}\exp(-r^2)d\sigma\left(\mathbf{s}_k\right)dr\\ &\approx\sum_{i=1}^{n_r}\sum_{j=1}^{n_s}\omega_{r,i}\omega_{s,j}h\left[r_i\mathbf{s}_j\right]\end{aligned} \quad (17)$$

Because the accuracy of the approximation using spherical-radial cubature rule depends on the degree of the cubature rule by giving the n_r and n_s, the fifth degree of the approximation of the spherical-radial cubature integral function is implemented. To achieve the requirements, the spherical-radial cubature rule has been separately calculated by using the radial rule, $\mathrm{S}(r)=\int h(r\mathbf{s})d\sigma(\mathbf{s})$, and the spherical rule, $\int S(r)r^{n-1}\mathrm{e}^{-r}\mathrm{d}r$, to represent ω_r, ω_s, r, and $h[r\mathbf{s}]$ as follows.

Generally, in the third degree, let n_r and n_s be 1 and $2n$, respectively. Then, the third-degree spherical-radial cubature rule integral function can be approximated by

$$\begin{aligned}I_{N,3}\left(h\right)&\approx\sum_{i=1}^{n_r}\sum_{j=1}^{n_s}\omega_{r,i}\omega_{s,j}h\left[r_i\mathbf{s}_j\right]\\ &=\frac{1}{2n}\sum_{i=1}^{2n}h\left(\sqrt{\mathbf{\Sigma}_{k|k-1}}\sqrt{n}\left[\mathbf{I}\right]_i+\hat{\mathbf{w}}_{k|k-1}\right)\end{aligned} \quad (18)$$

where $[\mathrm{I}]_i$ is the identity matrix that picks only i^{th} column, n is the size of the space of weight vector. In case of the fifth degree, we define n_r and n_s as 2 and

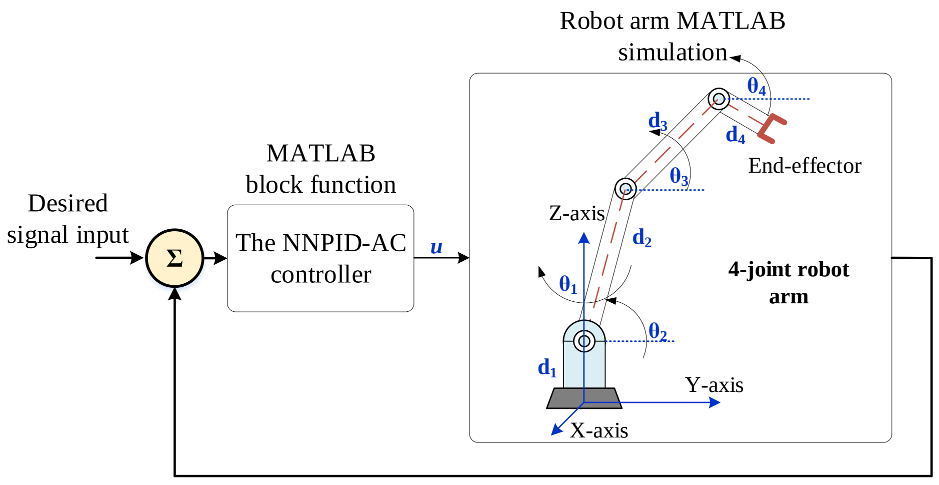

Figure. 3 The block diagram of the robot arm control system

$2n^2$, respectively. Then, the fifth-degree spherical-radial cubature rule integral function can be approximated by

$$
\begin{aligned}
I_{N,5}(h) &\approx \sum_{i=1}^{n_r}\sum_{j=1}^{n_s}\omega_{r,i}\omega_{s,j}h\left[r_i\mathbf{s}_j\right] \\
&= \frac{2}{n+2}h\left(\mathbf{w}_{k|k-1}\right) \\
&+ \frac{1}{(n+2)^2}\sum_{j=1}^{n(n-1)/2} h\left(\sqrt{\boldsymbol{\Sigma}_{k|k-1}}\sqrt{n+2}\left[\mathbf{I}_{N5}^{+}\right]_j+\hat{\mathbf{w}}_{k|k-1}\right) \\
&+ \frac{1}{(n+2)^2}\sum_{j=1}^{n(n-1)/2} h\left(-\sqrt{\boldsymbol{\Sigma}_{k|k-1}}\sqrt{n+2}\left[\mathbf{I}_{N5}^{+}\right]_j+\hat{\mathbf{w}}_{k|k-1}\right) \\
&+ \frac{4-n}{2(n+2)^2}\sum_{j=1}^{n(n-1)/2} h\left(\sqrt{\boldsymbol{\Sigma}_{k|k-1}}\sqrt{n+2}\left[\mathbf{I}_{N5}^{-}\right]_j+\hat{\mathbf{w}}_{k|k-1}\right) \\
&+ \frac{4-n}{2(n+2)^2}\sum_{j=1}^{n(n-1)/2} h\left(-\sqrt{\boldsymbol{\Sigma}_{k|k-1}}\sqrt{n+2}\left[\mathbf{I}_{N5}^{-}\right]_j+\hat{\mathbf{w}}_{k|k-1}\right)
\end{aligned}
\qquad (19)
$$

where $[I^{+}_{N5}]$ and $[I^{-}_{N5}]$ are given by

$$\left[\mathbf{I}_{N5}^{+}\right]_j = \sqrt{0.5}\left(\mathbf{I}_{ni}+\mathbf{I}_{nj}\right); ni < nj \qquad (20)$$

$$\left[\mathbf{I}_{N5}^{-}\right]_j = \sqrt{0.5}\left(\mathbf{I}_{ni}-\mathbf{I}_{nj}\right); ni < nj \qquad (21)$$

respectively, where n_i, $n_j = 1,2,3\ \ldots$

The state approximation from the spherical-radial cubature rule using the fifth degree is applied to the Gaussian integral equations from Eq. (10) to Eq. (14). All parameters will be used in the neural critic algorithm to calculate the local successor state of the control system.

4. Experimental setup

This section will give the explanation of how to set the robot arm control system demonstration by using the MATLAB/SIMULINK program as shown in Fig. 3, which divides into two-part; the controller block function simulation part and the robot arm block simulation part. We first create the robot arm block simulation from the MATLAB's library with some modifications of the physical and electrical parameters, including the no-load speed, the supply voltage, the armature inductance, the torque and the no-load current, which are set to 316 rad/s, 6v, 0.12 μH, 0.402 Nm, and 8 mA, respectively. Each joint of the robot arm simulations is set with movement sensors to detect coordination. The signals from the sensors will be sent back to the controller.

In case of the controller block simulation, we use the MATLAB block function to code the proposed algorithm. Besides, several types of controllers are also created using the MATLAB/SIMULINK block function for performance competitions, including the hybrid CKF-NNPID controller [19], the NNPID gradient-based controller [15], and the classical PID controller. At the beginning of the operation, all parameters such as the predicted plant output, error covariances, system error, are set to zero except the initial weights of those controllers. They must be initiated. The initial weights for each type of controllers are acquired by different methods as follows. First, the initial values of the hybrid CKF-NNPID controller can be obtained by a random method. The appropriated initial values will be generated after the end of the inner loop operation [19]. Next, the classical PID controller is obtained from the previous our work [15] that is tuned by MATLAB's library program. These values include

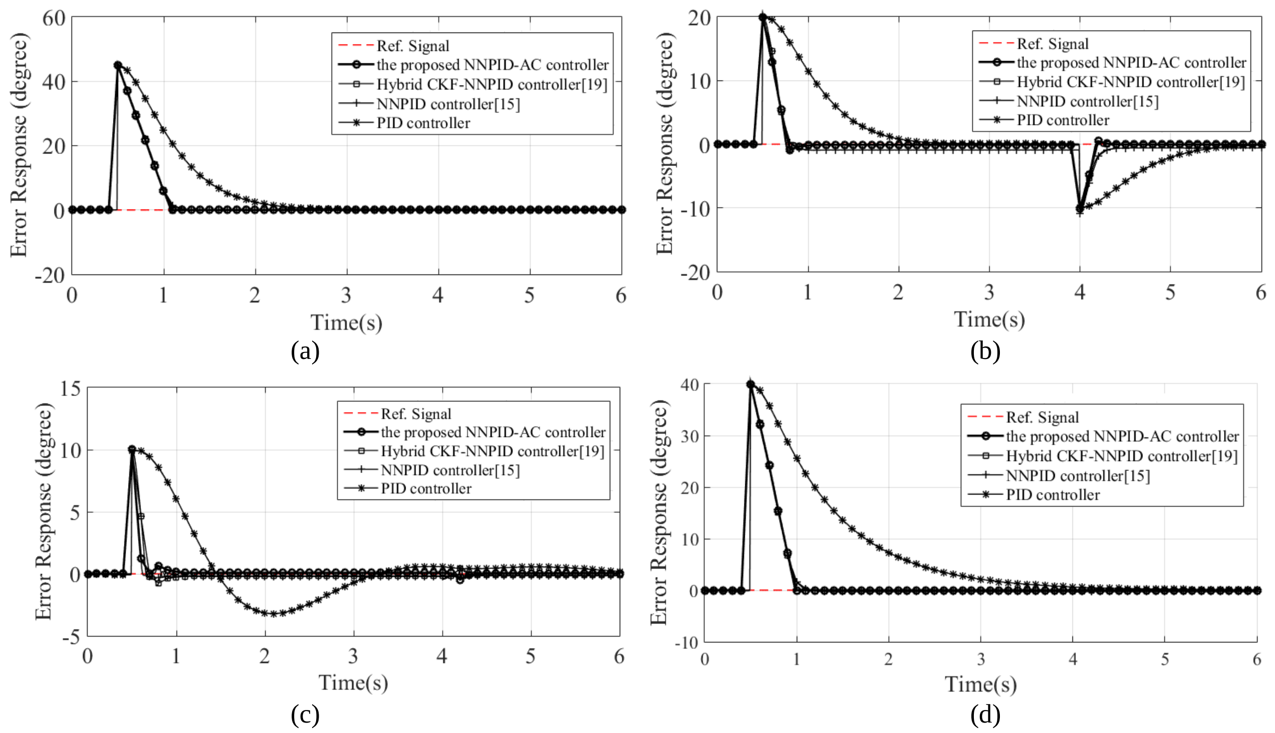

Figure. 4 The error responses of the step input signals of the system without load comparing with other controllers: (a) 1st joint, (b) 2nd joint, (c) 3rd joint, and (d) 4th joint

proportional gain (K_P), integral gain (K_I) and derivative gain (K_D), which are 25.0, 1.5 and -1.0, respectively. Finally, the NNPID gradient-based controller, the initial weights are taken from the tuning method similar to the classical PID controller. In case of the proposed NNPID-AC controller algorithm, the initial weights of the neural actor controller are initiated by random. Finally, all controllers must generate the control input signal driving the DC motor with range of 0 – 5 volts.

Furthermore, the simulations are set for various cases, including the control system without the load, the control system with the instantaneous noise, and the control system with the maximum load.

5. Results and discussion

From the results, the tests of the proposed NNPID-AC controller using the MATLAB simulations have been demonstrated. First, results of the robot arm control system without the load under the proposed NNPID-AC controller can potentially force the error back to zero, especially in the second joint and third joint that carry the torque more than other joints. In this case, the control input must generate the higher value of the voltage than the other joints to support the torque. On the contrary, it is a slightly different in the results of the first joint and fourth joint among the proposed controller, the Hybrid CKF-NNPID controller, and the PID-like controller because of the movements without the high torque. In case of the classical PID controller, the robot arm control system gives low performances in terms of transient response, percent overshoot and system stability.

Furthermore, to evaluate the system robustness, the results of the control system with instantaneous noise at the time of 2 second as shown in Fig. 5 that clearly prove the proposed NNPID-AC controller is the best. In other words, error responses of the control system under the proposed controller immediately converge to zero, while the classical PID controller takes a long time to converge back to the reference signal, especially in the second joint and third joint of the robot arm. In the normal operation, all computational intelligent controllers give similar results, which are better than the classical PID controller.

The results are significant when the maximum load is added to the control system. From Fig. 6, it shows that the angle response of the robot arm control system with the maximum load under the proposed controller is better than the other controllers, especially in the second joint and the third joint of the robot arm. In case of the third joint, all controllers except for the proposed NNPID-AC controller fail to maintain stability because of the high torque.

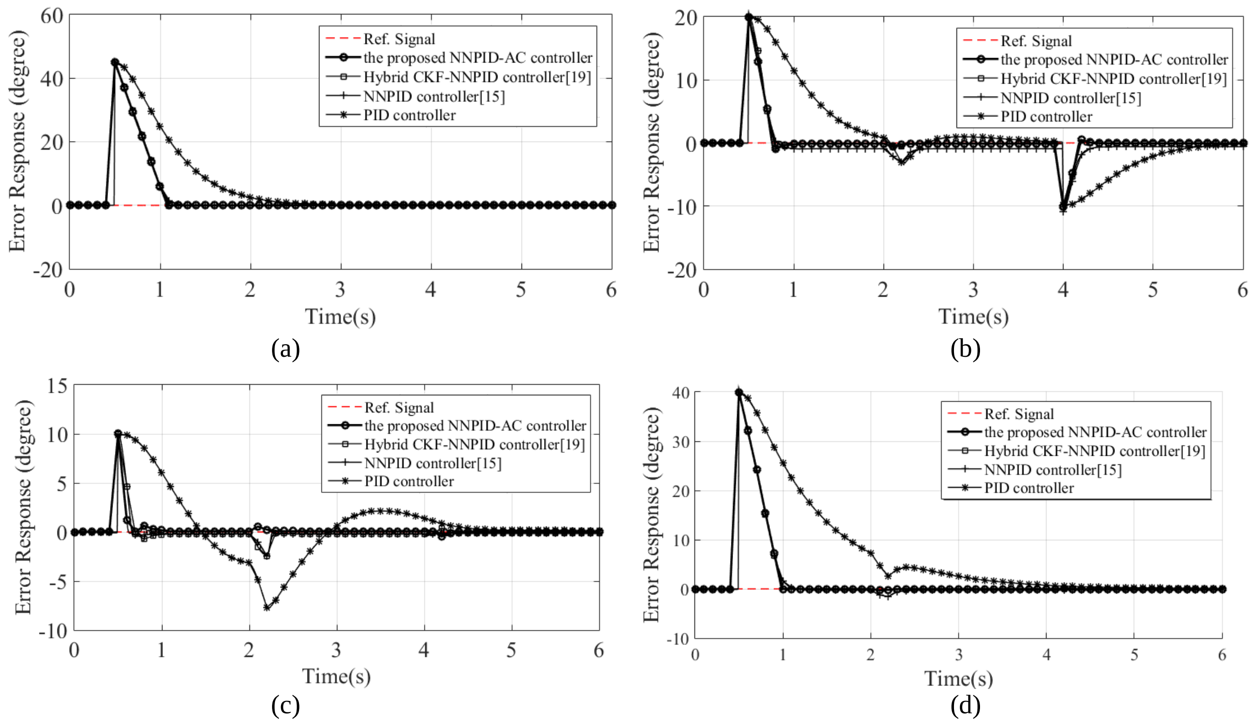

Figure. 5 The error responses of the step input signals of the system under disturbances at the time of 2 second comparing with other controllers: (a) 1st joint, (b) 2nd joint, (c) 3rd joint, and (d) 4th joint

Although the computational intelligent controllers use the iterative learning algorithm, the different learning algorithm yields different responses as shown in Table 1. It summarizes the angle response characteristics of the control system with the maximum load under the proposed controller comparing with other controllers, which are captured from Fig. 6 (b). From Table 1, it has been shown that the response characteristics under the proposed controller is the best in terms of the stead-state error, peak time, settling time and percent overshoot.

In each iteration, the proposed NNPID-AC controller emphasizes the next updated weights according to the desired control input from the model reference using the hybrid of the actor-critic learning algorithm and the square root cubature Kalman filter algorithm. In other words, the proposed controller uses the model reference and the system error to grade the output of the actor controller via the critic algorithm. While the Hybrid CKF-NNIPD controller uses the prediction algorithm of the next updated weights according to only the error of the system. Besides, the prediction algorithm of the proposed controller uses the fifth degree of the square root cubature Kalman filter algorithm for each step of the iterations, which is higher resolution than three degree. As a result, the learning algorithm of the proposed NNPID-AC controller can obtain more accuracy.

Table 1. Step response characteristic of the systems with the maximum load under various controllers

Type of controllers	**e_{ss} (degree)**	**T_P**	**T_S**	**%OS**
		(second)		
NNPID-AC	0.05	0.44	0.63	15.50
Hybrid CKF-NNPID	0.53	0.60	2.6	34.45
NNPID	3.13	0.51	1.29	22.85
PID	0.33	1.25	3.02	11.30

where e_{SS}, T_P, %OS and T_S are the steady-state error of the step response characteristics, peak time, percent overshoot and settling time, respectively.

Finally, in Fig. 7, the response error of the control system with the maximum load under the proposed controllers are compared with the other controllers. These results more clearly prove that the proposed NNPID-AC controller is the best controller.

6. Conclusion

A new controller design for the nonlinear control system using the actor-critic reinforcement learning algorithm has been proposed. In these studies, the weight of the actor controller is the important role in driving the robot arm along with the desired position. The critic algorithm using the fifth degree of the cubature Kalman filter has been selected to update the weight of the actor controller. By using the fifth degree of the cubature Kalman filter, the critic algorithm does not require searching through every action. Therefore, the proposed algorithm not only maintains the system stability, but also increases the performances and the speed. Furthermore, the robot arm MATLAB simulations also provides the control system with the load and noise to prove the robustness and fault tolerance. From the result, the proposed NNPID-AC controller gives the best performances either with or without noise disturbance and load.

In the future work, we will apply this algorithm to the real robot arm in real-world applications.

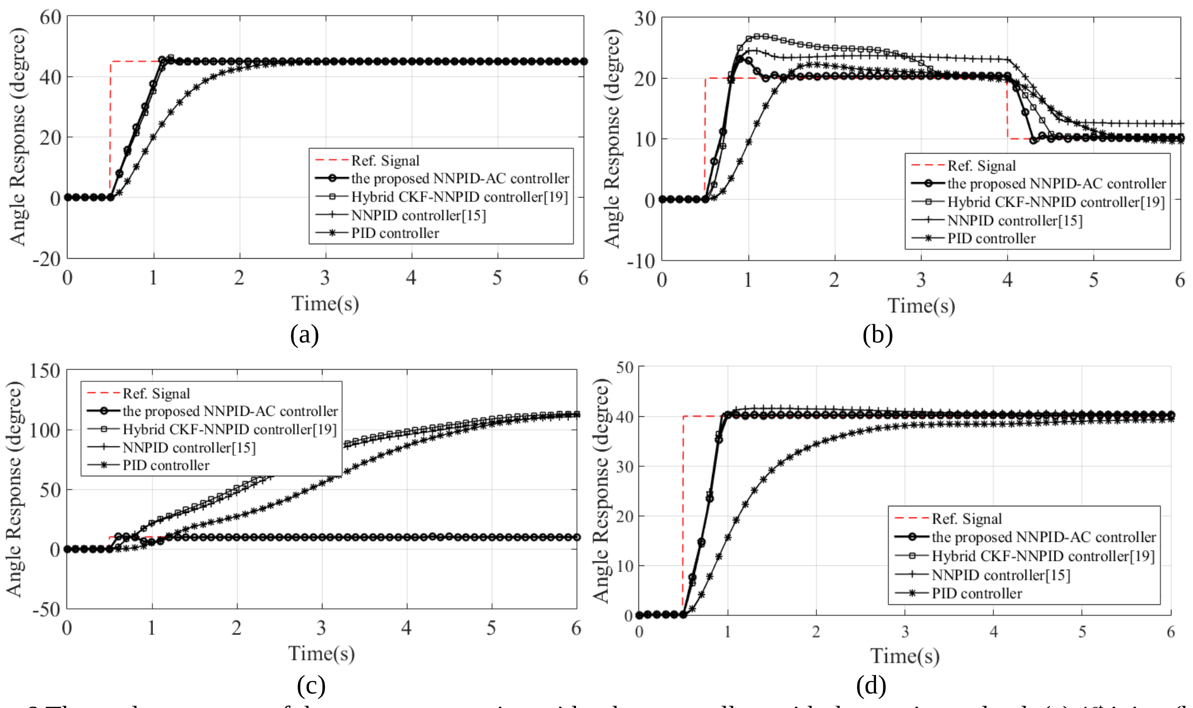

Figure. 6 The angle responses of the system comparing with other controllers with the maximum load: (a) 1st joint, (b) 2nd joint, (c) 3rd joint, and (d) 4th joint

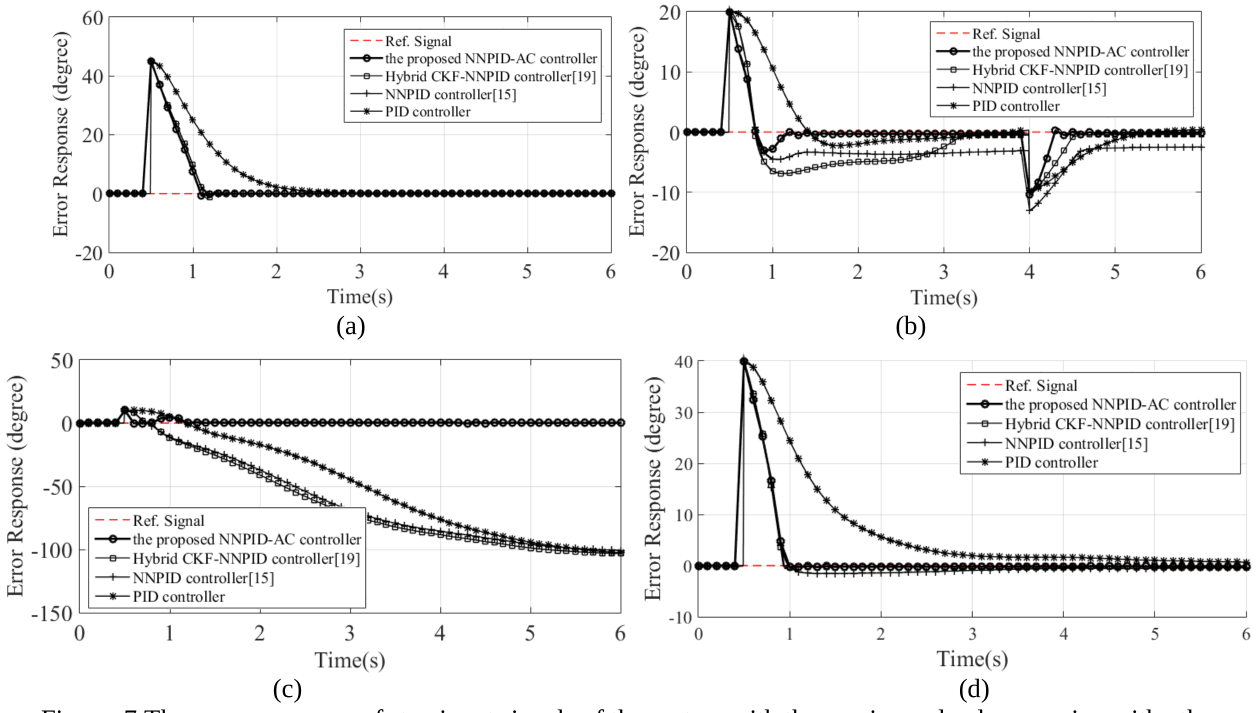

Figure. 7 The error responses of step input signals of the system with the maximum load comparing with other controllers: (a) 1st joint, (b) 2nd joint, (c) 3rd joint, and (d) 4th joint

References

[1] K.H. Ang, G. Chong, and Y. Li, "PID control system analysis, design, and technology", *IEEE Transactions on Control Systems Technology*, Vol. 13, No. 4, pp. 559-576, 2005.

[2] D. Rupp and L. Guzzella, "Adaptive internal model control with application to fueling control", *Control Engineering Practice*, Vol. 18, No. 8, pp. 873-881, 2010.

[3] C. C. Hang, K. J. Astrom, and W. K. Ho, "Refinements of the Ziegler-Nichols tuning formula", *IEE Proceedings D - Control Theory and Applications*, Vol. 138, No. 2, pp. 111-118, 1991.

[4] M.N. Anwar and S. Pan, "A frequency response model matching method for PID controller design for processes with dead-time", *ISA Transactions*, Vol. 55, pp. 175-187, 2015.

[5] C. Jiang, Y. Ma, and C. Wang, "PID controller parameters optimization of hydro-turbine governing systems using deterministic-chaotic-mutation evolutionary programming (DCMEP)", *Energy Conversion and Management*, Vol. 47, No. 9-10, pp. 1222-1230, 2006.

[6] A.B. Sharkawy, "Genetic fuzzy self-tuning PID controllers for antilock braking systems", *Engineering Applications of Artificial Intelligence*, Vol. 23, No. 7, pp. 1041-1052, 2010.

[7] I. Dounis, P. Kofinas, C. Alafodimos, and D. Tseles, "Adaptive fuzzy gain scheduling PID controller for maximum power point tracking of photovoltaic system", *Renewable Energy*, Vol. 60, pp. 202-214, 2013.

[8] B. Alagoz, A. Ates, and C. Yeroglu, "Auto-tuning of PID controller according to fractional-order reference model approximation for DC rotor control", *Mechatronics*, Vol. 23, No. 7, pp. 789-797, 2013.

[9] M. Moradi, "Self-tuning PID controller to three-axis stabilization of a satellite with unknown parameters", *International Journal of Non-Linear Mechanics*, Vol. 49, pp. 50-56, 2013.

[10] H. X. Li, L. Zhang, K. Y. Cai, and G. Chen, "An improved robust fuzzy-PID controller with optimal fuzzy reasoning", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 35, No. 6, pp. 1283-1294, 2005.

[11] A. Bouguerra, D. Saigaa, K. Kara, and S. Zeghlache, "Fault-Tolerant Lyapunov-Gain-Scheduled PID Control of a Quadrotor UAV", *International Journal of Intelligent Engineering and Systems*, Vol. 8, No. 2, pp. 1-6, 2015.

[12] Rubaai and P. Young, "EKF-Based PI-/PD-Like Fuzzy-Neural-Network Controller for Brushless Drives", *IEEE Transactions on Industry Applications*, Vol. 47, No. 6, pp. 2391-2401, 2011.

[13] C. Tsai, H. C. Huang, and S. C. Lin, "Adaptive Neural Network Control of a Self-Balancing Two-Wheeled Scooter", *IEEE Transactions on Industrial Electronics*, Vol. 57, No. 4, pp. 1420-1428, 2010.

[14] Idir, M. Kidouche, Y. Bensafia, K. Khettab, and S. Tadjer, "Speed Control of DC Motor Using PID and FOPID Controllers Based on Differential Evolution and PSO", *International Journal of Intelligent Engineering and Systems*, Vol. 11, No. 4, pp. 241–249, 2018.

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

[16] D.L. Yu, T. K. Chang, and D.W. Yu, "A stable self-learning PID control for multivariable time varying systems", *Control Engineering Practice*, Vol. 15, No. 12, pp. 1577-1587, 2007.

[17] A. Sento and Y. Kitjaidure, "Neural network controller based on PID using an extended Kalman filter algorithm for multi-variable non-linear control system", In: *Proc. of 2016 Eighth International Conf. on Advanced Computational Intelligence*, pp. 302-309, 2016.

[18] Arasaratnam and S. Haykin, "Cubature Kalman Filters", *IEEE Transactions on Automatic Control*, Vol. 54, No. 6, pp. 1254-1269, 2009.

[19] A. Sento and Y. Kitjaidure, "A hybrid CKF-NNPID controller for MIMO nonlinear control system", *ECTI Transactions on Computer and Information Technology,* Vol. 10, No. 2, pp. 176-184, 2017.

[20] Y. Chen, Q. Zhao, Z. An, P. Lv, and L. Zhao, "Distributed Multi-Target Tracking Based on the K-MTSCF Algorithm in Camera Networks", *IEEE Sensors Journal*, Vol. 16, No. 13, pp. 5481-5490, July, 2016.

[21] B. Jia and M. Xin, "Adaptive cubature Kalman filter with directional uncertainties [Correspondence]", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 52, No. 3, pp. 1477-1486, 2016.

[22] H. Modares, F. L. Lewis, and M. Naghibi-Sistani, "Adaptive Optimal Control of Unknown Constrained-Input Systems Using Policy Iteration and Neural Networks", *IEEE*

Transactions on Neural Networks and Learning Systems, Vol. 24, No. 10, pp. 1513-1525, 2013.

[23] B. Kiumarsi and F. L. Lewis, "Actor–Critic-Based Optimal Tracking for Partially Unknown Nonlinear Discrete-Time Systems", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 26, No. 1, pp. 140-151, 2015.

[24] J. Škach, B. Kiumarsi, F. L. Lewis, and O. Straka, "Actor-Critic Off-Policy Learning for Optimal Control of Multiple-Model Discrete-Time Systems", *IEEE Transactions on Cybernetics*, Vol. 48, No. 1, pp. 29-40, 2018.

[25] N. Bergman, *Recursive Bayesian Estimation Navigation and Tracking Applications*, Department of Electrical Engineering, Link¨oping University, Link¨oping, Sweden, 1999.

[26] R. E. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ. Republished, 2003.