HYPOTHESIS AND THEORY
published: 28 July 2020
doi: 10.3389/fnins.2020.00695

Hybrid Neural Network Cerebellar Model Articulation Controller Design for Non-linear Dynamic Time-Varying Plants

Tien-Loc Le[1,2], *Tuan-Tu Huynh*[2,3], *Sung-Kyung Hong*[1]* and *Chih-Min Lin*[3]

[1] *Faculty of Mechanical and Aerospace, Sejong University, Seoul, South Korea,* [2] *Department of Electrical Electronic and Mechanical Engineering, Lac Hong University, Bien Hoa, Vietnam,* [3] *Department of Electrical Engineering, Yuan Ze University, Taoyuan, Taiwan*

This study proposes a hybrid method to control dynamic time-varying plants that comprises a neural network controller and a cerebellar model articulation controller (CMAC). The neural-network controller reduces the range and quantity of the input. The cerebellar-model articulation controller is the main controller and is used to compute the final control output. The parameters for the structure of the proposed network are adjusted using adaptive laws, which are derived using the steepest-descent gradient approach and back-propagation algorithm. The Lyapunov stability theory is applied to guarantee system convergence. By using the proposed combination architecture, the designed CMAC structure is reduced, and it makes it easy to design the network size and the initial membership functions. Finally, numerical-simulation results demonstrate the effectiveness of the proposed method.

Keywords: neural network, cerebellar model articulation controller, time-varying plants, non-linear system, adaptive control

OPEN ACCESS

Edited by:
Vitor Engracia Valenti,
São Paulo State University, Brazil

Reviewed by:
Grienggrai Rajchakit,
Maejo University, Thailand
Ming-Feng Yeh,
Lunghwa University of Science and Technology, Taiwan

****Correspondence:***
Sung-Kyung Hong
skhong@sejong.ac.kr

Specialty section:
This article was submitted to Autonomic Neuroscience, a section of the journal Frontiers in Neuroscience

Received: *27 January 2020*
Accepted: *09 June 2020*
Published: *28 July 2020*

Citation:
Le T-L, Huynh T-T, Hong S-K and Lin C-M (2020) Hybrid Neural Network Cerebellar Model Articulation Controller Design for Non-linear Dynamic Time-Varying Plants. Front. Neurosci. 14:695. doi: 10.3389/fnins.2020.00695

INTRODUCTION

Nowadays, the control of non-linear systems is a topic that continues to attract many researchers because of its widespread applications. In many practical cases, the challenge of this topic is that its mathematical model is poorly known or uncertain (Liu et al., 2011). Furthermore, non-linear systems are susceptible to internal and external disturbances (Li et al., 2018). Therefore, in recent years, some studies have used neural networks (NNs) to approximate non-linear functions (Zhou and Zhang, 2015; Han, 2018). Some studies combined a neural network and other methods to achieve better control performance, such as proportional-integral-derivative (PID) NNs, fuzzy NNs, and sliding mode NNs (Zou et al., 2011; Zhou and Zhang, 2015; Lin and Le, 2017a; Zhao et al., 2018; Wang et al., 2019). Neural networks enable large-scale concurrent computing, processing, and adaptive weight adjustment, and they are simple and convenient (Prieto et al., 2016). Recently, many studies use NNs to address control problems, system identification and prediction problems. In 2013, Li et al. developed an optical-interference pattern-sensing method and neural-network classification for pretesting gap mura on thin-film transistor liquid crystal displays (Li et al., 2013). In 2017, Sun and Pan developed a reliable neural-network to control non-affine non-linear systems (Sun and Pan, 2017). In 2018, Wang et al. presented a memristor-based artificial neural network to predict house prices (Wang et al., 2018). However, neural networks require a considerable amount of computational resources, there is the risk of overfitting, and the architecture must be defined (Tu, 1996).

The concept of a cerebellar-model articulation controller (CMAC) was first proposed by Albus (1975). It is a type of neural network that uses a model of the mammalian cerebellum (associative memory). It addresses the problems of fast-growing size and the learning difficulties that are inherent to current neural networks. Several studies showed that, for applications that require online learning, CMACs perform better than simple neural networks (Lin and Chen, 2009; Guan et al., 2019). Since CMACs have a non-fully connected perceptron-like associative-memory network with overlapping receptive fields, they have fast learning performance, and its computation is simple. Contrarily, neural networks have a fully connected perceptron; therefore, all weights are updated during each learning cycle, so the learning capacity for a neural network is essentially global in nature and slow (Lin and Chen, 2009). The main advantages of CMACs over NNs, MLPs, and RBFNs are fast learning, simple computation, and good generalization capability (Lin et al., 2013). Recent studies have proposed some modified CMACs, such as function-link, self-organizing, and type-2 fuzzy CMACs that have better performance. In 2016, Lin et al. proposed a type-2 fuzzy CMAC for an adaptive filter (Lin et al., 2016). In 2017, Lin and Le used a wavelet CMAC to control non-linear systems (Lin and Le, 2017b). In 2018, Tsao et al. proposed the use of a deep CMAC for an adaptive noise-cancellation system (Tsao et al., 2018). A conventional CMAC also has some disadvantages, such as it is difficult to determine a suitable network size and to select the initial membership functions (MFs) to achieve the best performance (Lin and Chen, 2009). It is particularly difficult when the network has many inputs, and each input has a large range.

This study proposes a new method with a structure that includes a neural network connected in series with a CMAC. All inputs to the neural network reduce quantity and range. The outputs for the NN feed into the CMAC to compute the final outputs. This proposed network structure is referred to as a hybrid neural-network–CMAC (HNNCMAC). It is used to control dynamic time-varying plants. The motivation behind a cascade of two architectures was to allow for the inputs into the CMAC structure to be small, avoiding the difficulty in selecting a suitable network size and the initial membership functions. In the CMAC structure, the number of neurons in receptive-field spaces is increasing exponentially by the number of neurons in input space. Our proposed HNNCMAC controller using the NN to reduce the inputs for the CMAC, and then the structure of the modified CMAC in our proposed network will be smaller than the conventional CMAC. It is more effective when the number of inputs is large. In comparison with previous modified CMAC neural networks, as in Lin and Le (2017b) and Lin et al. (2018a,b), the proposed HNNCMAC has some advantages, such as small CMAC structure, and ease in designing network size and initial membership functions. The main contributions of this study are: (1) the successful design of an adaptive HNNCMAC system for the control of non-linear dynamic time-varying plants; (2) adaptive laws are derived using the steepest-descent gradient approach and a back-propagation algorithm; (3) input range and quantity in the proposed CMAC could be reduced by the NN pre-controller; (4) the stability of the proposed method is guaranteed by Lyapunov analysis; and (5) the method could be used for non-linear control problems, as proven by the results of numerical simulations.

The remaining sections of the paper are organized as follows. The design of the HNNCMAC is presented in section Methods. Section 3 presents the simulation results for controlling the dynamic time-varying plant. Section 4 provides the discussion. Finally, the conclusion is given in Section 5.

METHODS

HNNCMAC Structure

The structure of hybrid NNCMAC includes a neural network that is connected in series with a CMAC. The NN reduces the range and the quantity of the input, and the output from the NN becomes the input for the CMAC to compute the final control output. **Figure 1** shows the structure of HNNCMAC, which has seven spaces: input, hidden NN, output NN, association, receptive, weight-memory, and final-output spaces. These are described below.

(1) Input space I: There is no computation in this space. Input data from the dataset are fed into this space and directly transferred to the next space.

(2) Hidden NN space A: each node in this space performs a multiplication between vector input $I = \left[I_1,\, I_2, \ldots, I_{n_l}\right]^T$ and hidden NN weight matrix $h = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_{n_a} \end{bmatrix} = \begin{bmatrix} h_{11}, h_{12}, \ldots, h_{1n_l} \\ h_{21}, h_{22}, \ldots, h_{2n_l} \\ \vdots \quad \ddots \quad \vdots \\ h_{n_a1},\, h_{n_a2}, \ldots, h_{n_an_l} \end{bmatrix}$; after that, they are added with a bias $\alpha = \left[\alpha_1, \alpha_2, \ldots, \alpha_{n_a}\right]^T$, where n_a is the number of nodes in the hidden NN space and n_l is the number of nodes in the input space; h_{al} is the connecting weight from the a^{th}, $a = 1, ..., n_a$, neuron in space $\boldsymbol{A}$ to the l^{th}, $l = 1, ..., n_l$ neuron in space $\boldsymbol{I}$.

For example, in this space, the output from the x^{th} node is derived as

$$A_a = \left[h_{a1},\, h_{a2}, \ldots, h_{an_l}\right] \begin{bmatrix} I_1 \\ I_2 \\ \cdots \\ I_{n_l} \end{bmatrix} + \alpha_x \tag{1}$$

where α_a is the bias of the x^{th} neuron.

Then, the output from this space is expressed as $A = \left[A_1, A_2, \ldots, A_{n_a}\right]^T$.

(3) Output NN space B: This is the output from the neural-network space and it is the input for the CMAC. This layer performs a multiplication between the vector in the previous layer $A = \left[A_1, A_2, \ldots, A_{n_a}\right]^T$ and output NN weight matrix

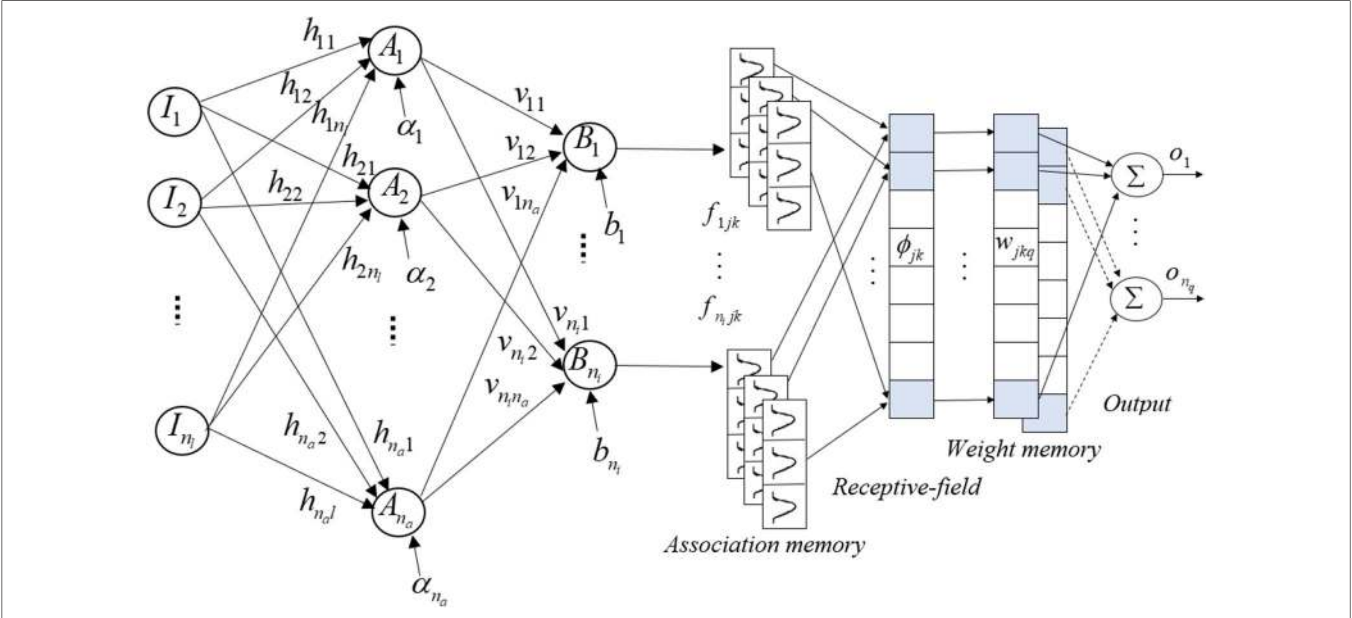

FIGURE 1 | Hybrid neural-network–cerebellar-model articulation-controller structure.

$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{n_i} \end{bmatrix} = \begin{bmatrix} v_{11}, v_{12}, \ldots, v_{1n_a} \\ v_{21}, v_{22}, \ldots, v_{2n_a} \\ \vdots \quad \ddots \quad \vdots \\ v_{n_i1}, v_{n_i2}, \ldots, v_{n_in_a} \end{bmatrix}$; then, it adds to a bias $b = \left[b_1,\ b_2, \ldots, b_{n_i}\right]^T$. To limit the input range for the CMAC, the final result in this space is a tangent sigmoid function. The output for the i^{th} node is

$$B_i = \text{tansig}(net_{B_i}) = \frac{e^{net_{B_i}} - e^{-net_{B_i}}}{e^{net_{B_i}} + e^{-net_{B_i}}}\ for\ i = 1, 2, \ldots, n_i \quad (2)$$

where $net_{B_i} = \left[v_{i1}, v_{i2}, \ldots, v_{in_a}\right] \begin{bmatrix} A_1 \\ A_2 \\ \ldots \\ A_{n_a} \end{bmatrix} + b_i$; v_{ia} is the connecting weight from the i^{th}, $i = 1, ..., n_i$, neuron in space $\boldsymbol{B}$ to the a^{th} neuron in space $\boldsymbol{A}$.

The output for this space is expressed as $B = \left[B_1, B_2, \ldots, B_{n_i}\right]^T$.

(4) Association space F: In this space, several elements are accumulated as a block. The membership grades in each block are calculated using input variables B_i from the previous space and the Gaussian MFs.

$$f_{ijk} = \exp\left(-\left(\frac{B_i - m_{ijk}}{\sigma_{ijk}}\right)^2\right)\ for\ j = 1, 2, \ldots, n_j\ and$$
$$k = 1, 2, \ldots, n_k \quad (3)$$

where m_{ijk} is the mean; σ_{ijk} is the variance of the k^{th} block in the j^{th} layer that corresponds to the i^{th} input variable; n_j is the number of layers; and n_k is the number of blocks. Therefore, the output from this space is the vector association.

(5) Receptive-field space ϕ : This layer performs the mapping that relates each location of F to generate the receptive-field vector:

$\phi = \left[\phi_{11}, ..., \phi_{1n_k}, ..., \phi_{n_j1}, ..., \ \phi_{n_jn_k}\right]^T \in \Re^{n_jn_k}$

where

$$\phi_{jk} = \prod_{i=1}^{n_i} f_{ijk}(B_i) \quad (4)$$

The mechanism for mapping 2D input is shown in **Figure 2**.

(6) Weight-memory space W: Each element of ϕ is mapped with a specific adjustable value for W that is expressed as

$$w_{jkq} = \begin{bmatrix} w_{111}, \ldots, w_{1n_k1}, ..., w_{n_j11}, ..., w_{n_jn_k1} \\ w_{112}, ..., w_{1n_k2}, ..., w_{n_j12}, ..., w_{n_jn_k2} \\ \vdots \quad \ddots \quad \vdots \\ w_{11n_q}, ..., w_{1n_kn_q}, ..., w_{n_j1n_q}, ..., w_{n_jn_kn_q} \end{bmatrix}^T \in \Re^{n_jn_kn_q} \quad (5)$$

where w_{jkq} is the connecting weight for the qth, $q = 1, ..., n_q$, final output and the receptive-field space for the jth layer and kth block.

(7) Final output space O: This space performs the product operation of receptive-field space ϕ and weight-memory space W to obtain the final output for the HNNCMAC, which is expressed as

$$u^q_{HNNCMAC} = o_q = w^T\phi = \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} w_{jkq}\phi_{jk} \quad (6)$$

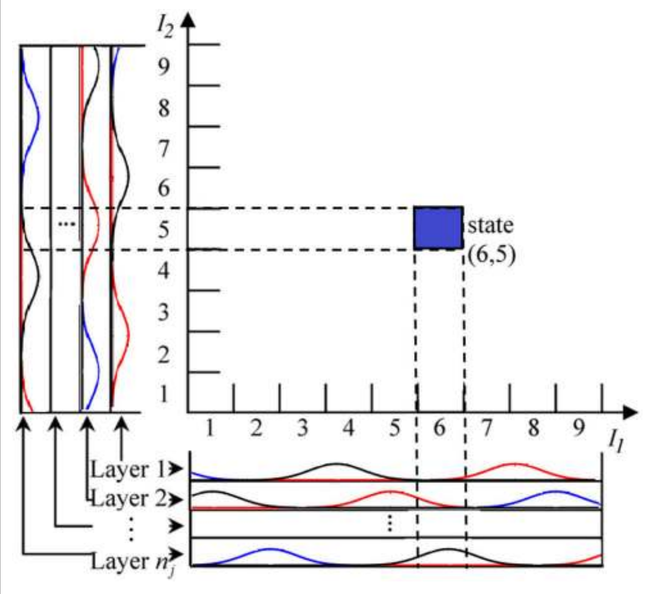

FIGURE 2 | CMAC mapping with Gaussian membership function.

The initial parameters for the HNNCMAC are chosen randomly and updated by some adaptive laws, which are derived using the steepest-descent gradient approach and a back-propagation algorithm, as described in the following section. The computational complexity using the Big-O notation is Big-O = O(T*[max($p_1, p_2, \ldots, p_{n_j}$) $+\prod_{j=1}^{n_j} p_j + n_l n_i n_a$]), where T is the running time, p_j is the number of membership functions in association space.

HNNCMAC Parameters—Learning Algorithm

The scheme for the HNNCMAC system is shown in **Figure 3**. The goal of control system is to generate control signal $\hat{u}_{HNNCMAC}(t)$, which forces the output of dynamic time-varying plant $y(t)$ to track reference signal $y_d(t)$. The flowchart of the HNNCMAC system is shown in **Figure 4**, in which input range and quantity in the proposed CMAC could be reduced by the NN pre-controller. Therefore, it can reduce the number of neurons in receptive-field spaces and the weight-memory space; then, the structure of CMAC can be significantly reduced.

The high-order sliding mode from Manceur et al. (2012) and Zheng et al. (2014) is used to improve the performance of the control system

$$\begin{aligned} s(t) &= \sum_{l=0}^{n-1} \frac{(n-1)!}{l!\,(n-l-1)!} \left(\frac{\partial}{\partial t}\right)^{n-l-1} \lambda^l e \\ &= e^{(n-1)} + (n-1)\lambda e^{(n-2)} + \frac{(n-1)(n-2)}{2}\lambda^2 e^{(n-3)} \\ &\quad \ldots + \lambda^{n-1} e \end{aligned} \tag{7}$$

where λ and n are the slope and the order of the sliding surface, respectively. Both λ and n are positive constants. Tracking error $e(t)$ is defined as:

$$e(t) = y_d(t) - y(t) \in \Re \tag{8}$$

where y_d and y are reference signal and system output, respectively.

Taking the derivative of Equation (7)

$$\begin{aligned} \dot{s}(t) &= e^{(n)} + (n-1)\lambda e^{(n-1)} + \frac{(n-1)(n-2)}{2}\lambda^2 e^{(n-2)} \\ &\quad \ldots + \lambda^{n-1} e \\ &= e^{(n)} + K^T e \end{aligned} \tag{9}$$

where $K = \left[(n-1)\lambda, \frac{(n-1)(n-2)}{2}\lambda^2, \ldots, \lambda^{n-1}\right]^T \in \Re^{n-1}$ is the positive gain vector and $e(t) = \left[e^{(n-1)}(t), e^{(n-2)}(t), \ldots, \dot{e}(t)\right]^T \in \Re^n$ is the tracking error vector.

If the values for n and λ correspond to the coefficients of a Hurwitz polynomial, then $\lim_{k\to\infty} e(t) = 0$.

The structure of the HNNCMAC has seven variables that are updated as: $w_{jkq}, m_{ijk}, \sigma_{ijk}, b_i, v_{ia}, \alpha_a$ and h_{al}. The Lyapunov cost function is chosen as $V(t) = \frac{1}{2}s^2(t)$, so $\dot{V}(t) = s(t)\dot{s}(t)$. An online learning gradient descent algorithm was used to minimize $\dot{V}(t)$. Therefore, online tuning laws for the HNNCMAC parameters are given by the following equations:

$$\hat{w}_{jkq}(k+1) = \hat{w}_{jkq}(k) + \Delta\hat{w}_{jkq} \tag{10}$$
$$\hat{m}_{ijk}(k+1) = \hat{m}_{ijk}(k) + \Delta\hat{m}_{ijk} \tag{11}$$
$$\hat{\sigma}_{ijk}(k+1) = \hat{\sigma}_{ijk}(k) + \Delta\hat{\sigma}_{ijk} \tag{12}$$
$$\hat{b}_i(k+1) = \hat{b}_i(k) + \Delta\hat{b}_i \tag{13}$$
$$\hat{v}_{ia}(k+1) = \hat{v}_{ia}(k) + \Delta\hat{v}_{ia} \tag{14}$$
$$\hat{\alpha}_a(k+1) = \hat{\alpha}_a(k) + \Delta\hat{\alpha}_a \tag{15}$$
$$\hat{h}_{al}(k+1) = \hat{h}_{al}(k) + \Delta\hat{h}_{al} \tag{16}$$

where $\hat{w}_{jkq}, \hat{m}_{ijk}, \hat{\sigma}_{ijk}, \hat{b}_i, \hat{v}_{ia}, \hat{\alpha}_a, \hat{h}_{al}$ are the estimation of the optimal values for parameters $w_{jkq}, m_{ijk}, \sigma_{ijk}, b_i, v_{ia}, \alpha_a, h_{al}$; and $\Delta\hat{w}_{jkq}, \Delta\hat{m}_{ijk}, \Delta\hat{\sigma}_{ijk}, \Delta\hat{b}_i, \Delta\hat{v}_{ia}, \Delta\hat{\alpha}_a, \Delta\hat{h}_{al}$ are the estimation of the optimal values for $\Delta w_{jkq}, \Delta m_{ijk}, \Delta\sigma_{ijk}, \Delta b_i, \Delta v_{ia}, \Delta\alpha_a, \Delta h_{al}$.

The updating term in Equations (10–16) is obtained by back-propagation by using the following chain rules:

$$\Delta\hat{w}_{jkq} = -\hat{\eta}_w \frac{\partial\dot{V}(t)}{\partial\hat{w}_{jkq}} = -\hat{\eta}_w \frac{\partial\dot{V}(t)}{\partial\hat{u}^q_{HNNCMAC}} \frac{\partial\hat{u}^q_{HNNCMAC}}{\partial\hat{w}_{jkq}} = \hat{\eta}_w s(t)\hat{\phi}_{jk} \tag{17}$$

$$\begin{aligned} \Delta\hat{m}_{ijk} &= -\hat{\eta}_m \frac{\partial\dot{V}(t)}{\partial\hat{m}_{ijk}} = -\hat{\eta}_m \frac{\partial\dot{V}(t)}{\partial\hat{u}^q_{HNNCMAC}} \frac{\partial\hat{u}^q_{HNNCMAC}}{\partial\hat{\phi}_{jk}} \frac{\partial\hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial\hat{m}_{ijk}} \\ &= -\hat{\eta}_m s(t)\,\hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})}{\hat{\sigma}^2_{ijk}} \end{aligned} \tag{18}$$

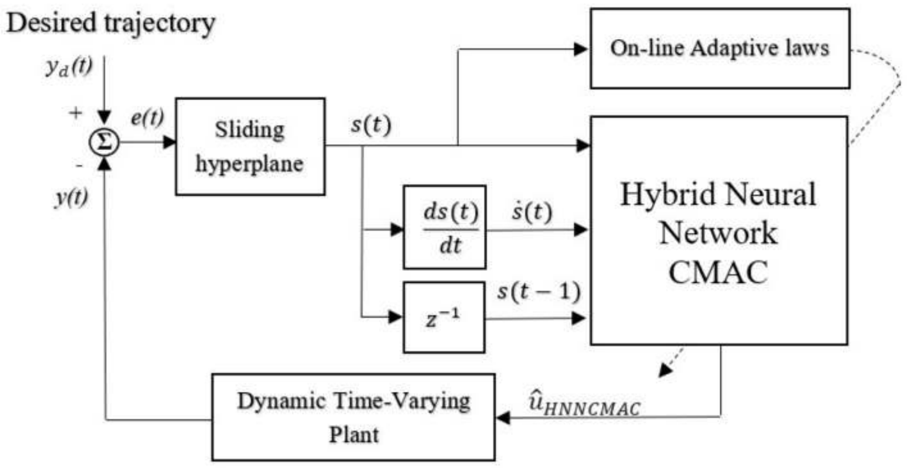

FIGURE 3 | Scheme for HNNCMAC system.

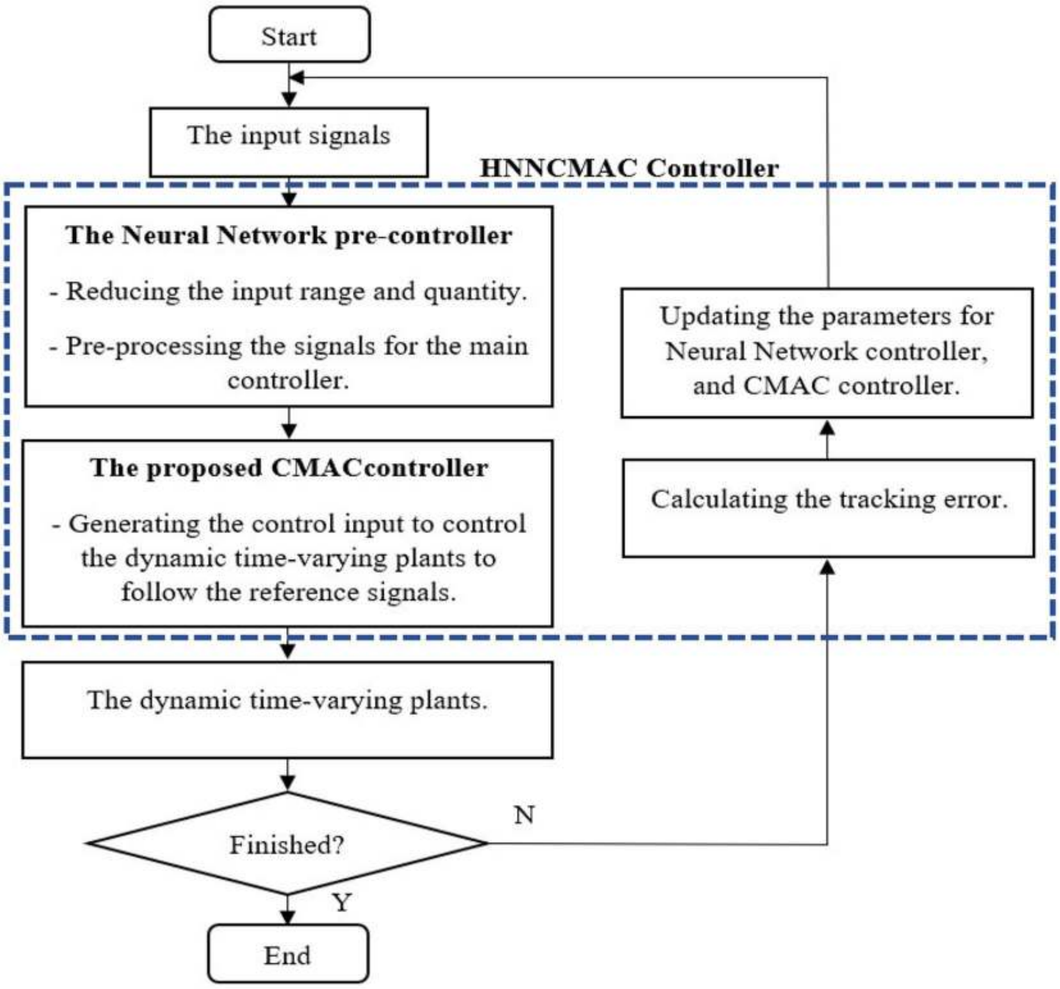
Start
The input signals
HNNCMAC Controller
The Neural Network pre-controller
- Reducing the input range and quantity.
- Pre-processing the signals for the main controller.
The proposed CMACcontroller
- Generating the control input to control the dynamic time-varying plants to follow the reference signals.
Updating the parameters for Neural Network controller, and CMAC controller.
Calculating the tracking error.
The dynamic time-varying plants.
Finished?
N
Y
End

FIGURE 4 | Flowchart of HNNCMAC system.

$$\Delta\hat{\sigma}_{ijk} = -\hat{\eta}_\sigma \frac{\partial \dot{V}(t)}{\partial \hat{\sigma}_{ijk}} = -\hat{\eta}_\sigma \frac{\partial \dot{V}(t)}{\partial \hat{u}^q_{HNNCMAC}} \frac{\partial \hat{u}^q_{HNNCMAC}}{\partial \hat{\phi}_{jk}} \frac{\partial \hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial \hat{\sigma}_{ijk}}$$
$$= -\hat{\eta}_\sigma s\,(t)\, \hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})^2}{\hat{\sigma}^3_{ijk}} \quad (19)$$

$$\Delta\hat{b}_i = -\hat{\eta}_b \frac{\partial \dot{V}(t)}{\partial \hat{b}_i}$$
$$= -\hat{\eta}_b \frac{\partial \dot{V}(t)}{\partial \hat{u}^q_{HNNCMAC}} \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\frac{\partial \hat{u}^q_{HNNCMAC}}{\partial \hat{\phi}_{jk}} \frac{\partial \hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial B_i} \frac{\partial B_i}{\partial net_{B_i}} \frac{\partial net_{B_i}}{\partial \hat{b}_i} \right)$$
$$= \hat{\eta}_b s\,(t) \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})}{\hat{\sigma}^2_{ijk}} \left(1 - B_i^2\right) \right) \quad (20)$$

$$\Delta\hat{v}_{ia} = -\hat{\eta}_v \frac{\partial \dot{V}(t)}{\partial \hat{v}_{ia}}$$
$$= -\hat{\eta}_v \frac{\partial \dot{V}(t)}{\partial \hat{u}^q_{HNNCMAC}} \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\frac{\partial \hat{u}^q_{HNNCMAC}}{\partial \hat{\phi}_{jk}} \frac{\partial \hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial B_i} \frac{\partial B_i}{\partial net_{B_i}} \frac{\partial net_{B_i}}{\partial \hat{v}_{ia}} \right)$$
$$= \hat{\eta}_v s\,(t) \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})}{\hat{\sigma}^2_{ijk}} \left(1 - B_i^2\right) A_i \right) \quad (21)$$

$$\Delta\hat{\alpha}_a = -\hat{\eta}_a \frac{\partial \dot{V}(t)}{\partial \hat{\alpha}_a}$$
$$= -\hat{\eta}_a \frac{\partial \dot{V}(t)}{\partial \hat{u}^q_{HNNCMAC}} \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\frac{\partial \hat{u}^q_{HNNCMAC}}{\partial \hat{\phi}_{jk}} \frac{\partial \hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial B_i} \frac{\partial B_i}{\partial net_{B_i}} \right)$$
$$\left(\left(\sum_{i=1}^{n_i} \frac{\partial net_{B_i}}{\partial A_a} \right) \frac{\partial A_a}{\partial \hat{\alpha}_a} \right)$$
$$= \hat{\eta}_a s\,(t) \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})}{\hat{\sigma}^2_{ijk}} \left(1 - B_i^2\right) \left(\sum_{i=1}^{n_i} v_{ia} \right) \right) \quad (22)$$

$$\Delta\hat{h}_{al} = -\hat{\eta}_h \frac{\partial \dot{V}(t)}{\partial \hat{h}_{al}}$$
$$= -\hat{\eta}_h \frac{\partial \dot{V}(t)}{\partial \hat{u}^q_{HNNCMAC}} \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\frac{\partial \hat{u}^q_{HNNCMAC}}{\partial \hat{\phi}_{jk}} \frac{\partial \hat{\phi}_{jk}}{\partial f_{ijk}} \frac{\partial f_{ijk}}{\partial B_i} \frac{\partial B_i}{\partial net_{B_i}} \right)$$
$$\left(\left(\sum_{i=1}^{n_i} \frac{\partial net_{B_i}}{\partial A_a} \right) \frac{\partial A_a}{\partial \hat{h}_{al}} \right)$$
$$= \hat{\eta}_h s\,(t) \sum_{j=1}^{n_j}\sum_{k=1}^{n_k} \left(\hat{w}_{jkq}\hat{\phi}_{jk} \frac{2(B_i - \hat{m}_{ijk})}{\hat{\sigma}^2_{ijk}} \left(1 - B_i^2\right) \left(\sum_{i=1}^{n_i} v_{ia} \right) I_l \right) \quad (23)$$

where $\hat{\eta}_m, \hat{\eta}_\sigma, \hat{\eta}_b, \hat{\eta}_v, \hat{\eta}_a, \hat{\eta}_h$ are the positive learning rates for the adaptive laws.

Using this online tuning parameter, the HNNCMAC can adjust the parameters online to achieve desired performance.

Proof of the algorithm convergence:
The Lyapunov cost function is defined as

$$V(t) = \frac{1}{2} s^2\,(t) \quad (24)$$

Therefore, the rate of change for Equation (24) is

$$\Delta\ V(t) = V(t+1) - V(t) = \frac{1}{2}\left[s^2(t+1) - s^2(t)\right] \quad (25)$$

By using the Taylor expansion, the difference in the sliding hyperplane is

$$s(t+1) = s(t) + \Delta s(t) \cong s(t) + \left[\frac{\partial s(t)}{\partial \hat{w}_{jkq}}\right] \Delta\hat{w}_{jkq} \quad (26)$$

From Equation (17), it can be seen that

$$\frac{\partial s(t)}{\partial \hat{w}_{jkq}} = -\hat{\phi}_{jk} \triangleq \xi \quad (27)$$

By using Equations (27) and (17), Equation (26) is rewritten as

$$s(t+1) = s(t) - \xi\left(\hat{\eta}_w s(t)\xi\right) = s(t)\left[1 - \hat{\eta}_w \xi^2\right] \quad (28)$$

By using Equation (28), Equation (25) is rewritten as

$$\Delta V(t) = \frac{1}{2} s^2(t)\left[\left(1 - \eta_w \xi^2\right)^2 - 1\right]$$
$$= \frac{1}{2} s^2(t)\left[\left(\hat{\eta}_w \xi^2\right)^2 - 2\hat{\eta}_w \xi^2\right]$$
$$= \frac{1}{2}\hat{\eta}_w s^2(t)\xi^2\left(\hat{\eta}_w \xi^2 - 2\right) \quad (29)$$

From Equation (29), if the learning rate $\hat{\eta}_w$ is $0 < \hat{\eta}_w < \frac{2}{\xi^2}$, then term $\Delta\ V(t)$ is negative, and Lyapunov function $V(t) > 0$. Therefore, the convergence of the system is guaranteed by Lyapunov stability. A similar method is used to prove the stability of learning rates $\hat{\eta}_m, \hat{\eta}_\sigma, \hat{\eta}_b, \hat{\eta}_v, \hat{\eta}_a, \hat{\eta}_h$.

SIMULATION OF RESULTS

In this section, the performance of the proposed HNNCMAC is investigated. Three examples in control of the dynamic time-varying plants are considered. The dynamic time-varying plants are the plants that contain the parameters varying with time.

Example 1: Controlling a dynamic time-varying plant borrowed from Narendra and Parthasarathy (1990) and Abiyev and Kaynak (2010), which is described by the difference equation

$$y\,(t) = f\left[y(t-1), y(t-2)\right] + u\,(t) + \varepsilon(t) + \Delta y(t) \quad (30)$$

where *u(t)* is the control signal from the proposed HNNCMAC; *y(t)*, *y(t-1)*, and *y(t-2)* are measurable plant output, one-step delayed plant output, and two-step delayed plant output, respectively; $\varepsilon(t) = 0.1\sin(\pi t)$ and $\Delta y(t) = 0.1y(t)$, respectively, denote the external disturbances and the system uncertainties;

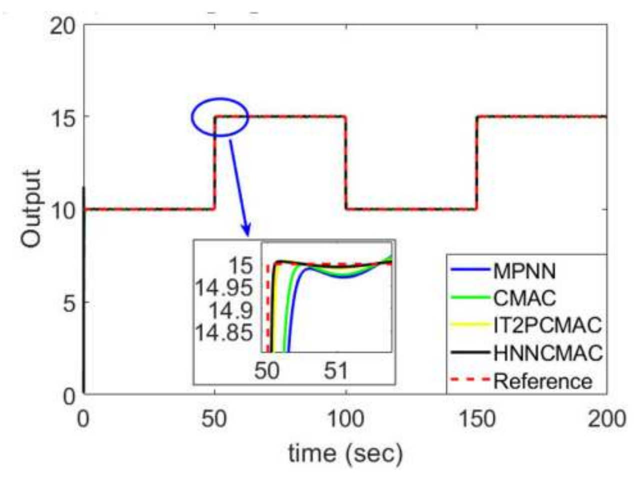

FIGURE 5 | Comparison of system outputs between proposed HNNCMAC and other control methods for Example 1.

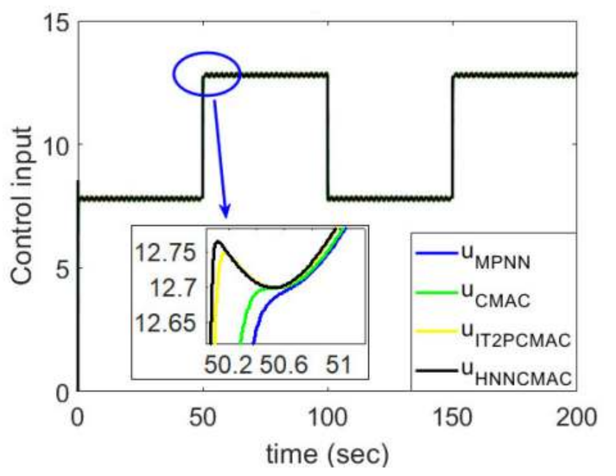

FIGURE 6 | Comparision of control signals between proposed HNNCMAC and other control methods for Example 1.

$f\left[y(t-1), y(t-2)\right]$ is the previous plant output function, which is given as $f\left[y(t-1), y(t-2)\right] = \frac{y(t-1)y(t-2)\left(y(t-1)+2.5\right)}{\left(1+y(t-1)^2+y(t-2)^2\right)}$

The desired trajectory signal $y_d(t)$ is given as

$$y_d(t) = \begin{cases} 10, & 0 < t \leq 50 \\ 15, & 50 < t \leq 100 \\ 10, & 100 < t \leq 150 \\ 15, & t > 150 \end{cases} \tag{31}$$

The desired trajectory signal and the system outputs for the dynamic time-varying plant are shown in **Figure 5**. Control signals and tracking errors are shown in **Figures 6**, **7**, respectively. These results show that the HNNCMAC allows a time-varying plant to follow a specified trajectory signal. In terms of the performance of the control system, **Table 1** shows a comparison of the root mean square error (RMSE) for the proposed method and other methods.

Example 2: Controlling a dynamic time-varying plant borrowed from Zhang et al. (1998) and Abiyev and Kaynak (2010), which is described by the difference equation

$$y(t) = f\left[y(t-1), y(t-2)\right] + b_0(t)\, u(t) + \varepsilon(t) + \Delta y(t) \tag{32}$$

where *u(t)* is the control signal from the proposed HNNCMAC; *y(t)*, *y(t-1)*, and *y(t-2)* are measurable plant output, one-step delayed plant output, and two-step delayed plant output, respectively; $\varepsilon(t) = 0.1\sin(\pi t)$ and $\Delta y(t) = 0.1y(t)$, respectively, denote the external disturbances and the system uncertainties; $f\left[y(t-1), y(t-2)\right]$ is the previous plant output function, which is given as $f\left[y(t-1), y(t-2)\right] = b_1(t)\, y(t-1) + b_2(t)\, y(t-2)$; $b_0(t)$, $b_1(t)$, and $b_2(t)$ are the time-varying function, which are given as $b_0(t) = -\frac{t^2}{1+a_1(t)t+a_2(t)t^2}$; $b_1(t) = \frac{2+a_1(t)t}{1+a_1(t)t+a_2(t)t^2}$; $b_2(t) = -\frac{1}{1+a_1(t)t+a_2(t)t^2}$; $a_1(t)$ and $a_2(t)$ are the time-varying plant parameters, which are given as

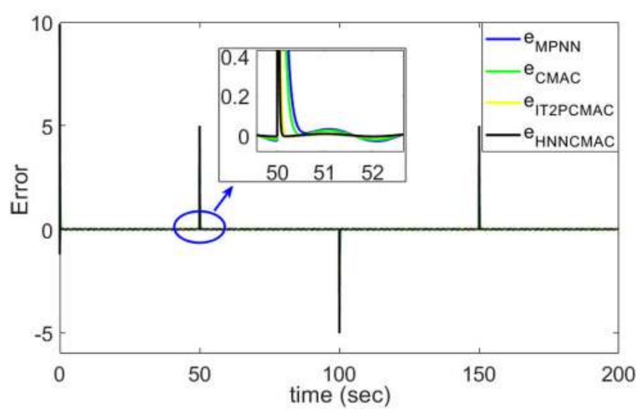

FIGURE 7 | Comparison of tracking errors between proposed HNNCMAC and other control methods for Example 1.

$$a_1(t) = \frac{0.1t}{t+1};\; a_2(t) = \begin{cases} 0.3, & 0 \leq t < 40 \\ 0.1, & 40 \leq t < 60 \\ 0.6, & 60 \leq t < 85 \\ 0.3, & t > 85 \end{cases} \tag{33}$$

The desired trajectory signal is given as

$$y_d(t) = \begin{cases} 10, & 0 < t \leq 25 \\ 15, & 25 < t \leq 50 \\ 10, & 50 < t \leq 75 \\ 15, & t > 75 \end{cases} \tag{34}$$

Figure 8 shows the change in the time-varying parameters. The desired trajectory signal and the outputs for the dynamic time-varying plant are shown in **Figure 9**. Control signals and tracking errors are shown in **Figures 10**, **11**, respectively. Simulation

TABLE 1 | Comparison results in root mean square error (RMSE) of control time-varying systems.

Control method	Computation time (s)	Example 1	Example 2	Example 3 (Square)	Example 3 (Sinusoidal)
MPNN	0.0158	0.1878	0.8679	1.7629	0.4901
Conventional CMAC	0.0327	0.1692	0.8407	1.7141	0.4552
T2TSKFNS	0.0416	0.1469	0.7395	*N*	*N*
IT2PCMAC	0.0382	0.1408	0.7683	1.5297	0.4225
HNNCMAC (proposed controller)	0.0254	0.1215	0.6708	1.1644	0.3498

N: the articles did not show those results. Note: MPNN, multilayer perceptron neural network; T2TSKFNS, Takagi–Sugeno–Kang fuzzy neural system; IT2PCMAC, interval type-2 Petri CMAC.

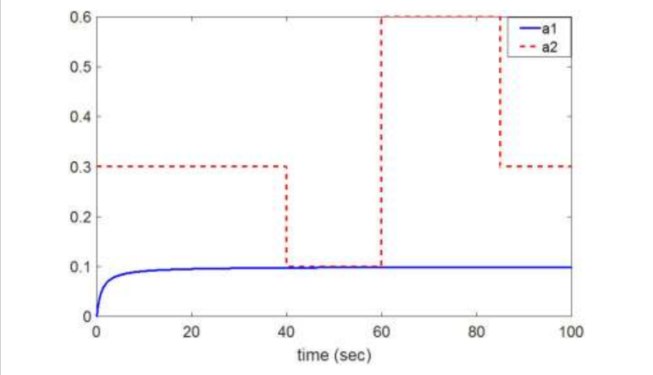

FIGURE 8 | Change of time-varying parameters a_1 and a_2.

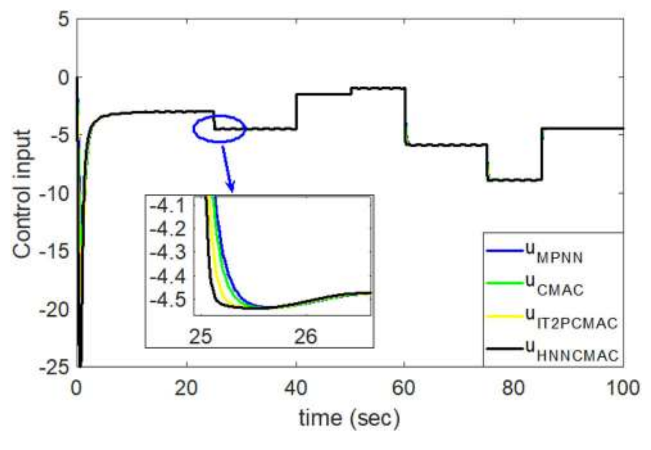

FIGURE 10 | Comparision of control signals between proposed HNNCMAC and other control methods for Example 2.

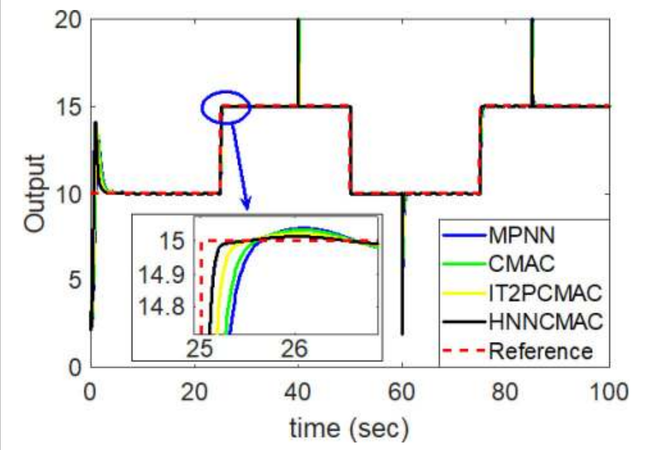

FIGURE 9 | Comparison of system outputs between proposed HNNCMAC and other control methods for Example 2.

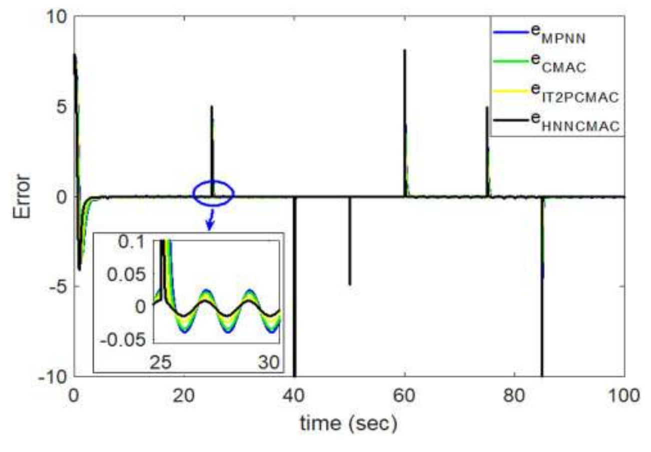

FIGURE 11 | Comparison of tracking errors between proposed HNNCMAC and other control methods for Example 2.

results showed that the HNNCMAC allows a time-varying plant to follow the reference signal, even if there are abrupt changes in parameters a_1 and a_2. **Table 1** shows a comparison of the RMSE for the proposed method and other methods.

Example 3: Controlling a dynamic time-varying plant to follow variable frequency signals.

This example uses the same dynamic time-varying plant that is described in Example 2. The desired trajectory signal is the variable frequency signal:

$$y_{d1}(t) = 5 * square(2\pi t k_t) \tag{35}$$

and

$$y_{d2}(t) = 5 * sin(2\pi t k_t) \quad (36)$$

where *square* and *sin* are the square function and the sinusoidal function, respectively, and k_t is the parameter for changing the signal frequency, which changes by time as follows:

$$k_t(t) = \begin{cases} 0.1, & 0 \le t < 40 \\ 0.5, & 40 \le t < 60 \\ 0.75, & 60 \le t < 85 \\ 1.0, & t > 85 \end{cases} \quad (37)$$

By using the square signal with varying frequency in Equation (35) as the desired trajectory, the reference signal and the system outputs for the time-varying plant are shown in **Figure 12**. The control signals and tracking errors for this case are shown in **Figures 13**, **14**, respectively. **Figure 15** shows the reference signals and system outputs for the time-varying plant when the desired trajectory is the sinusoidal signal with varying frequency in Equation (36). The control signals are plotted in **Figure 16**, and the tracking errors are plotted in **Figure 17**. Simulation results for the sinusoidal function reference showed that, at the beginning of the control process, the proposed controller could control the system well, but as frequency increases with time, as well as when the time-varying plant parameters suddenly change, the tracking error also rises due to the controller needing time to adapt to these changes. As shown in **Figures 7**, **11**, **14**, **17**, there were some rapid variation errors at the time the reference signals or the time-varying plant parameters suddenly changed. However, our proposed controller showed better ability to adapt to these changes, and the tracking error using our proposed HNNCMAC could quickly converge better than other control methods can. The external disturbances and the system uncertainties in this case are chosen as $\varepsilon(t) = 0.8\sin(\pi t)$ and $\Delta y(t) = 0.3y(t)$, respectively. A comparison of the RMSE for the following variable-frequency signal is shown in **Table 1**.

DISCUSSIONS

For this control problems, the HNNCMAC structure had three neurons in the input space, 10 neurons in the hidden space,

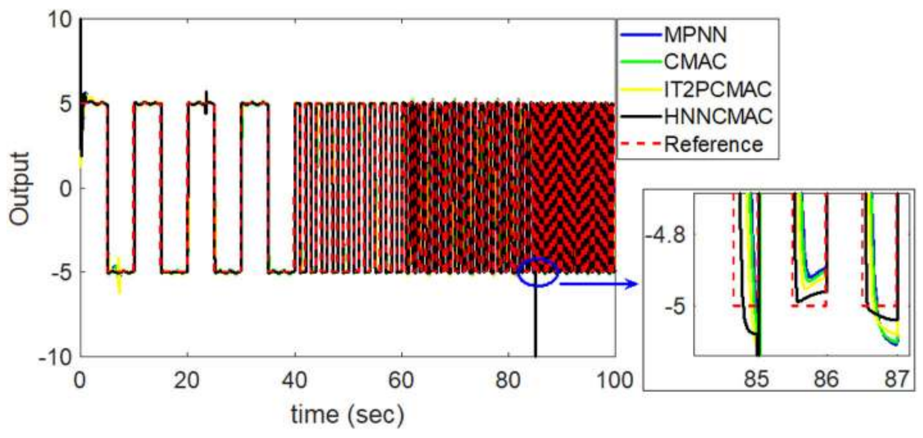

FIGURE 12 | Comparison of system outputs between proposed HNNCMAC and other control methods for square signal reference with varying frequency.

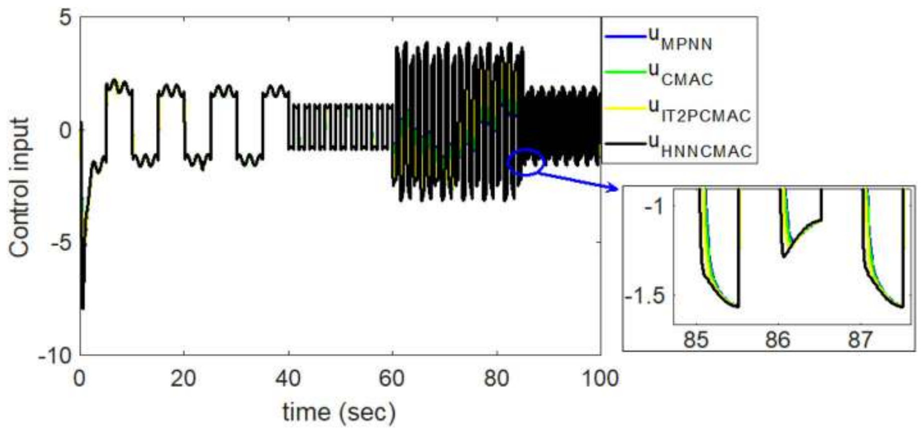

FIGURE 13 | Comparision of control signals between proposed HNNCMAC and other control methods for square signal reference with varying frequency.

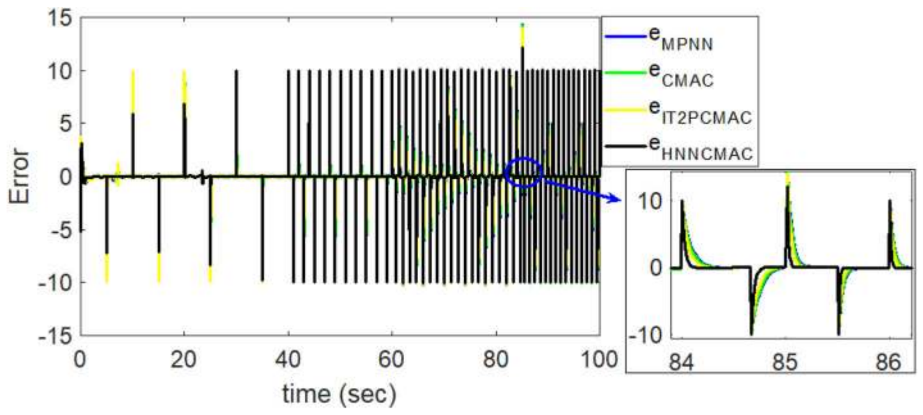

FIGURE 14 | Comparison of tracking errors between proposed HNNCMAC and other control methods for square signal reference with varying frequency.

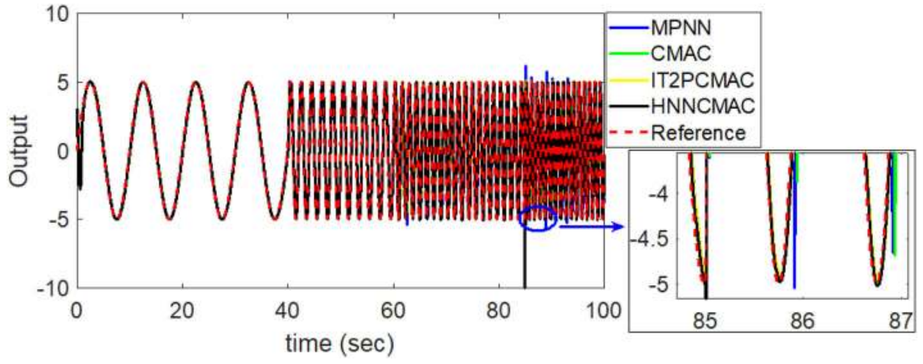

FIGURE 15 | Comparison of system outputs between proposed HNNCMAC and other control methods for sinusoidal signal reference with varying frequency.

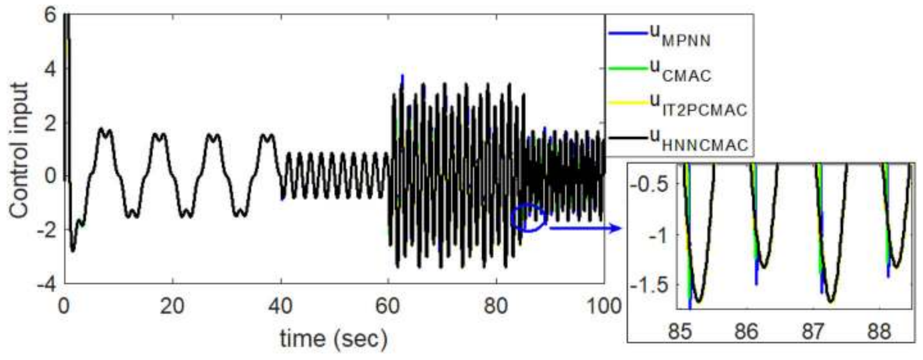

FIGURE 16 | Comparision of control signals between proposed HNNCMAC and other control methods for sinusoidal signal reference with varying frequency.

and two neurons in output space NN. The association space had two layers, each of which with five Gaussian membership functions. The input for the HNNCMAC control system was the output from the sliding hyperplane, its one-step delayed, and its derivatives, $s(t)$, $s(t-1)$, and $\dot{s}(t)$. Term $s(t-1)$ is used to obtain more information about the time-varying plants. The initial parameters for the Gaussian function were $m_{11k} = m_{21k} = m_{31k} = [-0.5 \quad -0.3 \quad 0 \quad 0.3 \quad 0.5]$, $m_{12k} = m_{22k} = m_{32k} = [-0.45 \quad -0.35 \quad -0.5 \quad 0.25 \quad 0.45]$, and $\sigma_{ijk} = 0.4$. The parameters for the sliding surface were $n = 3$ and $\lambda = 0.2$. All learning rates were 0.01, and sampling time was 0.01 s. Using the adaptation laws in Equations (10–23), the controller parameters can be updated online to adapt to the changes in the control system. The examples have demonstrated that our

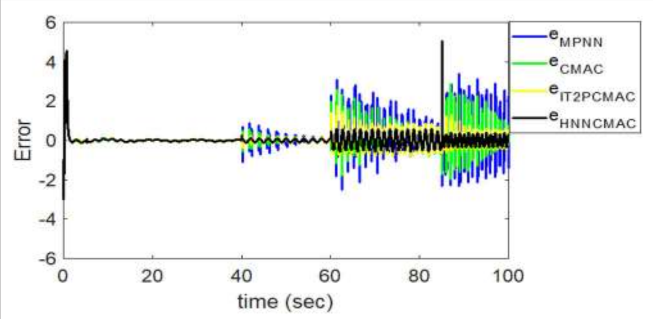

FIGURE 17 | Comparison of tracking errors between proposed HNNCMAC and other control methods for sinusoidal signal reference with varying frequency.

proposed controller can address well the external disturbances and the system uncertainties. The convergence of our proposed controller is guaranteed by Lyapunov stability analysis approach in Equation (29). The average RMSE for all examples between the proposed HNNCMAC, the multilayer perceptron NN (MPNN), the conventional CMAC, the interval type-2 Petri CMAC (IT2PCMAC) (Le et al., 2019), and the type-2 Takagi–Sugeno–Kang fuzzy neural system (T2TSKFNS) (Abiyev and Kaynak, 2010) are shown in **Table 1**. It is obvious that the proposed controller was using the NN to reduce the inputs for the CMAC; then, the structure of the modified CMAC in our proposed network would be smaller than that of a conventional CMAC. It is more effective when the number of inputs is large. **Table 1** shows that the proposed controller has a small computation time than a conventional CMAC, due to our modified CMAC structure was using the NN pre-controller to reduce the computation complexity of the CMAC. Moreover, the NN output used the tangent sigmoid function to limit the output from [−1 1]. Therefore, it is easy to design the network size and the initial membership functions in our modified CMAC controller. As shown in **Table 1**, the proposed HNNCMAC algorithm could achieve better control performance with the smallest RMSE than other controllers could. In **Appendix A, Tables A–D** show analysis of the difference between our proposed controller and other controllers using the *t*-Test statistical approach. In all examples, statistical results showed that the *P-value* was lower than the alpha level ($\alpha = 0.05$). Thus, we can conclude that the RMSE results of our proposed controller had statistically significant difference with other controllers. Therefore, the superiority of the proposed controller was illustrated. Some real-world applications, which have large inputs, can apply this proposed network to reduce the network structure such as medical diagnosis problems, classification problems, image processing problems, etc. Choosing the parameters for the sliding surface affects much of the control performance. This study used the try-and-error approach to obtain suitable parameters. Further studies should investigate the estimation method to estimate these parameters to achieve better control performance.

CONCLUSIONS

This paper proposed an HNNCMAC that is used to control a non-linear dynamic time-varying plant. The main contributions of this study are that it demonstrated a method to control a non-linear dynamic time-varying plant; the HNNCMAC structure uses adaptive laws to adjust parameters online; input range and quantity in the proposed CMAC can be reduced by the NN pre-controller, and it makes it easy to design network size and initial membership functions; the stability of the proposed method is guaranteed by Lyapunov analysis and the numerical-simulation results for controlling a time-varying plant, showing the superiority of the proposed method over existing methods. Moreover, our proposed controller is simple to design and implement, and can be applied to other fields such as system identification, classification, and prediction. Our future work will apply the optimal algorithm to optimize parameters in the sliding surface and learning rates in adaptive laws to achieve better control performance.

DATA AVAILABILITY STATEMENT

All datasets generated/analyzed for this study are included in the article/**Supplementary Material**.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

FUNDING

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2019-2018-0-01423) supervised by the IITP (Institute for Information & communications Technology Promotion).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnins.2020.00695/full#supplementary-material

REFERENCES

Abiyev, R. H., and Kaynak, O. (2010). Type 2 fuzzy neural structure for identification and control of time-varying plants. *IEEE Trans. Ind. Electron.* 57, 4147–4159. doi: 10.1109/TIE.2010.2043036

Albus, J. S. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *J. Dyn. Syst. Meas. Control* 97, 220–227. doi: 10.1115/1.3426922

Guan, J. S., Hong, S. J., Kang, S. B., Zeng, Y., Sun, Y., and Lin, C. M. (2019). Robust adaptive recurrent cerebellar model neural network for non-linear system based on GPSO. *Front. Neurosci.* 13:390. doi: 10.3389/fnins.2019.00390

Han, Y. Q. (2018). Adaptive tracking control of nonlinear systems with dynamic uncertainties using neural network. *Int. J. Syst. Sci.* 49, 1391–1402. doi: 10.1080/00207721.2018.1453955

Le, T. L., Lin, C.M., and Huynh, T. T. (2019). Interval type-2 Petri CMAC design for 4D chaotic system", in: *2019 International Conference on System Science and Engineering (ICSSE)* (Dong Hoi: IEEE), 420–424. doi: 10.1109/ICSSE.2019.8823251

Li, D. P., Liu, Y. J., Tong, S., Chen, C. P., and Li, D. J. (2018). Neural networks-based adaptive control for nonlinear state constrained systems with input delay. *IEEE Trans. Cybernetics* 49, 1249–1258. doi: 10.1109/TCYB.2018.2799683

Li, T. Y., Tsai, J. Z., Chang, R. S., Ho, L. W., and Yang, C. F. (2013). Pretest gap mura on TFT LCDs using the optical interference pattern sensing method and neural network classification. *IEEE Trans. Ind. Electron.* 60, 3976–3982. doi: 10.1109/TIE.2012.2207658

Lin, C. M., and Chen, T. Y. (2009). Self-organizing CMAC control for a class of MIMO uncertain nonlinear systems. *IEEE Trans. Neural Netw.* 20, 1377–1384. doi: 10.1109/TNN.2009.2013852

Lin, C. M., Huynh, T. T., and Le, T. L. (2018a). Adaptive TOPSIS fuzzy CMAC back-stepping control system design for nonlinear systems. Comput. 23, 6947–6966. doi: 10.1007/s00500-018-3333-4

Lin, C. M., La, V. H., and Le, T. L. (2018b). DC–DC converters design using a type-2 wavelet fuzzy cerebellar model articulation controller. *Neural Comput. Appl.* 32, 2217–2229. doi: 10.1007/s00521-018-3755-z

Lin, C. M., and Le, T. L. (2017a). PSO-self-organizing interval type-2 fuzzy neural network for antilock braking systems. *Int. J. Fuzzy Syst.* 19, 1362–1374. doi: 10.1007/s40815-017-0301-6

Lin, C. M., and Le, T. L. (2017b). WCMAC-based control system design for nonlinear systems using PSO. *J. Intell. Fuzzy Syst.* 33, 807–818. doi: 10.3233/JIFS-161999

Lin, C. M., Lin, M. H., and Yeh, R. G. (2013). Synchronization of unified chaotic system via adaptive wavelet cerebellar model articulation controller. *Neural Comput. Appl.* 23, 965–973. doi: 10.1007/s00521-012-1021-3

Lin, C. M., Yang, M. S., Chao, F., Hu, X. M., and Zhang, J. (2016). Adaptive filter design using type-2 fuzzy cerebellar model articulation controller. *IEEE Trans. Neural Netw. Learn. Syst.* 27, 2084–2094. doi: 10.1109/TNNLS.2015.2491305

Liu, K., Sun, D., Yao, Y., and Balakrishnan, V. (2011). "A new approach to stabilization of uncertain nonlinear systems", in *2011 IEEE International Symposium on Computer-Aided Control System Design (CACSD)* (Denver, CO: IEEE), 228–233. doi: 10.1109/CACSD.2011.6044540

Manceur, M., Essounbouli, N., and Hamzaoui, A. (2012). Second-order sliding fuzzy interval type-2 control for an uncertain system with real application. *IEEE Trans. Fuzzy Syst.* 20, 262–275. doi: 10.1109/TFUZZ.2011.2172948

Narendra, K. S., and Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. Neural Netw.* 1, 4–27. doi: 10.1109/72.80202

Prieto, A., Prieto, B., Ortigosa, E. M., Ros, E., Pelayo, F., Ortega, J., et al. (2016). Neural networks: an overview of early research, current frameworks and new challenges. *Neurocomputing* 214, 242–268. doi: 10.1016/j.neucom.2016.06.014

Sun, T., and Pan, Y. (2017). Adaptive control for nonaffine nonlinear systems using reliable neural network approximation. *IEEE Access* 5, 23657–23662. doi: 10.1109/ACCESS.2017.2763628

Tsao, Y., Chu, H. C., Fang, S. H., Lee, J., and Lin, C. M. (2018). Adaptive noise cancellation using deep cerebellar model articulation controller. *IEEE Access* 6, 37395–37402. doi: 10.1109/ACCESS.2018.2827699

Tu, J. V. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *J. Clin. Epidemiol.* 49, 1225–1231. doi: 10.1016/S0895-4356(96)00002-9

Wang, J., Hu, S., Zhan, X., Luo, Q., Yu, Q., Liu, Z., et al. (2018). Predicting house price with a memristor-based artificial neural network. *IEEE Access* 6, 16523–16528. doi: 10.1109/ACCESS.2018.2814065

Wang, S., Chen, S., Ji, W., and Liu, K. (2019). Quantized sampled-data control for TS fuzzy system using discontinuous LKF approach. *Front. Neurosci.* 13:372. doi: 10.3389/fnins.2019.00372

Zhang, C. J., Shao, C., and Chai, T. Y. (1998). Indirect adaptive control for a class of linear time-varying plants. *IEE Proc. Control Theory Appl.* 145, 141–149. doi: 10.1049/ip-cta:19981847

Zhao, J., Lin, C. M., and Chao, F. (2018). Wavelet fuzzy brain emotional learning control system design for MIMO uncertain nonlinear systems. *Front. Neurosci.* 12:918. doi: 10.3389/fnins.2018.00918

Zheng, E. H., Xiong, J. J., and Luo, J. L. (2014). Second order sliding mode control for a quadrotor UAV. *ISA Transactions* 53, 1350–1356. doi: 10.1016/j.isatra.2014.03.010

Zhou, M., and Zhang, Q. (2015). Hysteresis model of magnetically controlled shape memory alloy based on a PID neural network. *IEEE Trans. Mag.* 51, 7301504–7301507. doi: 10.1109/TMAG.2015.2434933

Zou, A. M., Kumar, K. D., Hou, Z. G., and Liu, X. (2011). Finite-time attitude tracking control for spacecraft using terminal sliding mode and chebyshev neural network. *IEEE Trans. Syst. Man Cybernetics B* 41, 950–963. doi: 10.1109/TSMCB.2010.2101592

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.