DOI: https://doi.org/10.33103/uot.ijccce.19.2.3

Design a Different Structures Controller for Controlled Systems Using a Spiking Neural Network

Mohammed Y. Hassan [1], Ahmed Abduljabbar Mahmood [2]

[1,2]*Control and Systems Engineering Department, University of Technology, Baghdad, Iraq*
60003@uotechnology.edu.iq, *ahmed_aazz2006@yahoo.com*

Abstract— *The design and simulation of the Spiking Neural Network (SNN) are proposed in this paper to control a plant without and with load. The proposed controller is performed using Spike Response Model. SNNs are more powerful than conventional artificial neural networks since they use fewer nodes to solve the same problem. The proposed controller is implemented using SNN to work with different structures as P, PI, PD or PID like to control linear and nonlinear models. This controller is designed in discrete form and has three inputs (error, integral of error and derivative of error) and has one output. The type of controller, number of hidden nodes, and number of synapses are set using external inputs. Sampling time is set according to the controlled model. Social-Spider Optimization algorithm is applied for learning the weights of the SNN layers. The proposed controller is tested with different linear and nonlinear models and different reference signals. Simulation results proved the efficiency of the suggested controller to reach accurate responses with minimum Mean Squared Error, small structure and minimum number of epochs under no load and load conditions.*

Index Terms— *Spiking Neural Network, Social-Spider Optimization, P/PI/PD/PID-like, Intelligent control.*

I. INTRODUCTION

Most biological neurons communicate by sending short and sudden increase in voltage (short pulses) between the connections of the neurons. The pulse that is generated is known as a “spike”. It refers to short and transient nature. With the coming spikes to the neurons, the neurons are influenced and can generate the spike when their membrane potential (voltage or state) becomes larger than or equal to the threshold value. These neurons are known as spiking neurons and the networks that include these neurons are called Spiking Neural Networks (SNNs) [1]. SNN is the third generation of artificial neural networks (ANNs). Computationally, the SNNs are more powerful and need fewer nodes to solve the same problem than conventional artificial neural networks [2]. Therefore; SNN controller can be achieved with a small network size. There are many approaches for training the SNNs; such as SpikeProp also called Error Back Propagation (BP) algorithm [3], Spikelm (applying the Levenberg-Marquardt (LM) algorithm to SNN) [3], Remote Supervised Method (ReSuMe) [4], Resilient Propagation (RProp) and QuickProp [5]. Also evolution algorithms like; Particle Swarm Optimization (PSO) or Genetic algorithm (GA) can be used as a training algorithm to find the optimal solutions. SSO is better than PSO and GA by classification performance that uses various estimation indicators and in wrapper-based

Received 26 Sep 2018; Accepted 4 Dec 2018

feature selection [6]. In recent years, a memorable attention in SNNs has grown and many types of researches studied SNNs as a controller.

Cao et al. (2015) [7] designed a three-layer SNN controller for target tracking of the autonomous robots. They used the SRM. The Hebbian learning was used to update the weights. The target information, which was supplied to the CCD cameras, ultrasonic sensors and encoders, is encoded in the shape of spike trains and integrated signals. The controller makes the robot follow the target trajectory without collision with the environment. Clawson et al. (2016) [8] presented a flapping micro robot (RoboBee) SNN controller that shuts the circle between the locally available sensors and actuators, by methods of a leaky integrate-and-fire SNNs, that are adjusted in flight utilizing a reward modulated Hebbian plasticity mechanism. Hernandez et al. in 2017 [9] developed a neuromorphic system on a Spartan-6 FPGA board to generate locomotion patterns for three different legged robots (biped, quadruped, and hexapod) based on spiking neurons. It implements the locomotion patterns for different legged robots in FPGA chip. Antonietti et al. in 2018 [10] developed the SNN to control NAO Robot. It was a simplified model of the neurophysiological cerebellar network, including neural connectivity, plasticity mechanisms, and simplified neuron dynamics. The Pavlovian conditioning protocol tested the model capability to learn the temporal associations between two stimuli provided to the humanoid NAO robot. The developed platform is a real neurorobot which is able to operate in a dynamic and noisy environment, dealing with a simple learning motor task.

In this paper, the design of a SNN as a controller to work with different structures (P, PI, PD or PID like), is presented to control linear and nonlinear models, different types of reference inputs, different sampling times and different load conditions. The type of controller, number of hidden nodes, and number of synapses are set using external inputs. The best weights of the SNN are obtained using SSO algorithm. The criterion used in obtaining the best weights is the minimum MSE.

This paper is structured as follows: Section II represents the model, structure, coding and decoding method of feed-forward SNN. Also, section III represents the SSO algorithm, while Section IV represents the design and simulation for the proposed controller. Finally, the conclusions are given in section V.

II. SPIKING NEURAL NETWORK

The third generation of artificial neural networks (ANN) is called a SNN. It is introduced by Wolfgang Mass in 1997 [2]. SNN has the same structure of the ANN but has synapses between each neuron layers. Fig. 1 shows the neuron model and the synapses in feed-forward SNN.

A. Neuron Model

There are many models for spiking neuron, for example, integrate and fire neuron, Izhikevich model, Spike Response Model, Fitzhugh model and Hodgkin-Huxley (HH) model [11]. The most accurate model among these models is HH, but it's very complex. SRM is simple due to its arithmetical simplicity and it becomes the most widely contributed model [12]. Therefore, SRM is used as a SNN model in this paper. The SRM was introduced by Gerstner in 1995. It described the relationship between input spikes and the internal state variable. The feed-forward SNN structure shown in Fig. (1-A) consists of three layers, input layer (H), hidden layer (I), and output layer (J). Each layer is connected with other layers by a set of synapses and every synaptic has various weights and delay times, as shown in Fig. (1-B). There are different forms of SRM and according to the choice

Received 26 Sep 2018; Accepted 4 Dec 2018

of suitable Spike Response Functions (SRF). This model can be regulated to watch the dynamics of different spiking neurons [13]. It is supposed that every neuron can generate only one spike as a maximum through the interval time of the simulation and it fires when the interior status variable is greater than or equal to the threshold value.

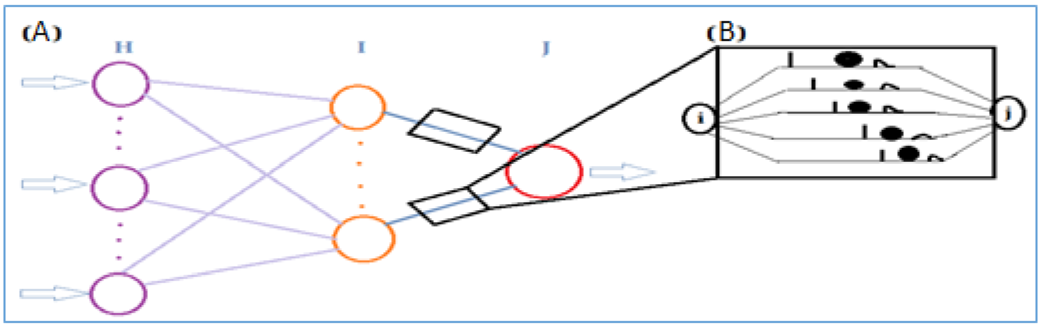


FIG.1: STRUCTURE OF SNN: (A) FEED-FORWARD SNN, (B) CONNECTION OF MULTIPLE DELAYED SYNAPTIC TERMINALS.

Also, it is assumed that pre-synaptic neuron (i) fires a spike at time (t_i) and the (K^{th}) synapse transfers that spike into the post-synaptic neuron at $(t_i + d^k)$. (d^k) represents the delay correlated with the (K^{th}) synapses. The interior status (membrane potential) $(m_j(t))$ of the post-synaptic neuron was represented by the following SRF [13]:

$$m_j(t) = \sum_{i=1}^{N} \sum_{k=1}^{K} w_{ij}^k \varepsilon(t - t_i - d^k) \tag{1}$$

, where (N) is the pre-synaptic neurons number, (K) is the synapses number, (t) is the current time and $(\mathrm{w}_{\mathrm{ij}}^{\mathrm{k}})$ is the coefficient of the weight between pre-synaptic neuron (i) and post-synaptic neuron (j) for the (K^{th}) synapse. It is trained using SSO algorithm afterwards. The SRF $(\varepsilon(t))$ termed the influence of the pre-synaptic neuron potential on the postsynaptic neuron potential [12]. There are different mathematical forms of SRF such as the hyperbolic tangent function that is used in this paper, as below [14]:

$$\varepsilon(t) = \tanh\frac{\mathrm{t}}{\tau} \tag{2}$$

and its derivative:

$$\frac{d\varepsilon}{dx} = \frac{1}{\tau}\left(1 - tanh^2\left(\frac{t}{\tau}\right)\right) \tag{3}$$

, where (τ) refers to the time constant that determines growth and decay in this SRF.

B. Coding and Decoding with SRM

The feed-forward SNN comprises three layers, input layer (H), hidden layer (I), and output layer (J). The transmitting of the data through SRM of SNN is described in the following steps [14]:

1. Data pass to the input layer neurons is encoded into spikes time and is computed using the following equation [12]:

$$t_h^f(I_{in}) = T_{max} - round\left(T_{min} + \frac{(I_{in}-I_{min})(T_{max}-T_{min})}{I_{max}-I_{min}}\right) \tag{4}$$

where $\left(t_h^f\right)$ is the firing time for input neurons, (T_{max}) is the maximum spike time and (T_{min}) is the minimum spike time. (I_{max}) is the largest value in the input data, (I_{min}) is the lowest value in the input data, and (I_{in}) is the present value of the input data.
2. Every hidden layer neuron is tested if it is spiked or not spiked. At most, every neuron can spike once just amid the time interval, when its membrane potential (m_i) is greater than or equals the threshold (ϑ). When neuron (i) is spiked at time (t_i^f), the algorithm

Received 26 Sep 2018; Accepted 4 Dec 2018

goes to the next neuron (i+1) in the current layer. Otherwise, the membrane potential ($m_i(t)$) is computed by the following equation [13]:

$$m_i(t) = \sum_{h=1}^{Hn} \sum_{k=1}^{Dn} w_{hi}^k(t)\varepsilon(t - t_h^f - d^k) \tag{5}$$

, where (H_n) is the number of the i/p layer neurons, ($\mathrm{D_n}$) is the number of sub-synaptic, (w_{hi}^k) is the weight between neurons in the input layer and hidden layer, ($\mathrm{d^k}$) denotes to the delay of the connection and $\varepsilon(\mathrm{t} - \mathrm{t_h^f} - \mathrm{d^k})$ is the response function. When ($m_i(t)$) exceeds the threshold value at instant (t), the neuron (i) will be prohibited from spiking more in the remaining time interval (T), otherwise it will be reset in the next instant ($t +$ 1). When the neurons of the hidden layer are finished, the same algorithm is repeated to the output layer (j), but in this case the spikes of the hidden layer will be the inputs to the neurons of the output layer. The membrane potential ($m_j(t)$) is calculated as [12]:

$$m_j(t) = \sum_{i=1}^{In} \sum_{k=1}^{Dn} w_{ij}^k \varepsilon(t - t_i^f - d^k) \tag{6}$$

, where (I_n) is the neurons number in the hidden layer, (t_i^f) is the firing time for hidden layer (i) and (t_j^f) is the firing time of output layer (j).

3. Finally, the output information of SNN is converted from time to real number using equation (7), which is extracted from the coding equation (4):

$$RI\left(t_j^f\right) = \frac{\left(T_{max} - t_j^f - T_{min}\right) * (I_{max} - I_{min})}{(T_{max} - T_{min})} \tag{7}$$

, where $RI\left(t_j^f\right)$ is the real information for the firing time of the output layer (j). After completing the feed-forward calculations, the error is computed using Mean Squares Error (MSE) [12]:

$$MSE = \sum_{j \in J} \left(t_j^f - t_j^d\right)^2 \tag{8}$$

, where (t_j^d) is training target spike time for output neuron (j).

III. SOCIAL SPIDER OPTIMIZATION (SSO) ALGORITHM

Social spider optimization (SSO) is a swarm optimization algorithm that was introduced by Erik Cuevas et al. in 2013. The SSO algorithm mimics the social attitude of the spiders colony in nature [6]. Colonies are created fundamentally from two elements. The first one is the spiders and the second one is a communal web. It represents the web by the domain of the searching field and representing the problem solutions [15]. SSO algorithm is applied in many optimization problems, but it is different from other optimization methods. It applies simple mathematical operators and does not require complex operators where its time and space complexity are not expensive [16]. This algorithm mimics the collective behavior of social spiders and works according to the two different search operators, which are known as females and males [17]. Spiders are divided by gender (male and female) and each one has a different behavior in the colony. The number of females (N_f) is chosen at random within the range (65% - 90%) of the number of spiders [6]:

$$N_f = \text{floor}\ ((0.9 - \text{rand} * 0.25) * N_s) \tag{9}$$

, while the number of male spiders (N_m) is calculated using the following equation [6]:

$$N_m = N_s - N_f \tag{10}$$

, where (N_s) is the number of spiders, (floor) maps a real number to an integer number and (rand) is a number that is generated randomly between $[0,1]$.

The total population denoted by (S) is divided into two groups; the female group denoted by (F) that consists of female members ($F = f1, f2, f3, \dots, fN_f$) and the male group that

Received 26 Sep 2018; Accepted 4 Dec 2018

consists of male members ($M = m1, m2, m3, \ldots., mN_m$) and denoted by (M). The position of the female spider (fi) with the position of the male spider (mi) is selected randomly between the initial parameters (p^{low}) and (p^{high})that represent the lower bound and upper bound, respectively [6].

$$f_{i,j}^{t} = p_j^{low} + rand * \left(p_j^{high} - p_j^{low}\right) \tag{11}$$

$$m_{i,j}^{t} = p_j^{low} + rand * \left(p_j^{high} - p_j^{low}\right) \tag{12}$$

, where (i) is the parameter index and (j) is the individual index. Now, it must calculate the fitness evaluation using the fitness function for each spider (WI). It represents the solution (in this paper (WI) consists of (w_{hi}^{k}) and (w_{ij}^{k})). This calculation is done using [6]:

$$W_{I=} \frac{J(_{Si}) - worst_s}{best_s - worst_s} \tag{13}$$

, where (Si) is the spider position and ($J(_{Si})$) is the fitness value that has been obtained for this position (Si), the values of bests and worsts are the minimum and the maximum values for the solution in the population (minim value problem) as defined in equations (14) and (15), respectively [6]:

$$best_s = min_{i\epsilon[1,2,3,\ldots,N]}(J_{(si)}) \tag{14}$$

$$worst_s = max_{i\epsilon[1,2,3,\ldots,N]}(J_{(si)}) \tag{15}$$

The mechanism used to send information through the colony members is communal web. This information is encoded as small vibrations. They depend on the distance and weight of each spider which has created them. The vibrations spotted by the spider (i) from spider (j) are defined as [6]:

$$V_i\, b_{i,j} = w_j \,.\, e^{-\,d_{i,j}^2} \tag{16}$$

, where ($d_{i,j}$) represents the distance (Euclidian) between the spiders (i) and spiders (j), where $d_{i,j}$=$\|s_i - s_j\|$. Any pair of spiders has three relations of vibrations between them. These vibrations are used with the SSO algorithm. The first one is denoted by ($Vibci$) that represents the vibrations of the spider (i) with the member $c(s_c)$ that is the nearest individual to the spider (i) with a higher weight. ($Vibci$) is described in equation (17) [6]. The second one is denoted by ($Vibbi$) which represents the vibrations of the spider (i) with the member $b(s_b)$ that is the best fitness (weight) value of the total population (S). ($Vibbi$) is described in equation (18) [6]. The third one is denoted by ($Vibfi$), which represents the vibrations of the spider (i) with the nearest female member $f(s_f)$, as can be described in equation (19) [6].

$$Vibci{=}w_c \,.\, e^{-\,d_{i,c}^2} \tag{17}$$

$$Vibbi{=}w_b \,.\, e^{-\,d_{i,b}^2} \tag{18}$$

$$Vibfi{=}w_f \,.\, e^{-\,d_{i,f}^2} \tag{19}$$

In the female cooperative process, female spiders offer attraction or disfavor to the other spiders regardless of their gender. The vibrations rely on the spider distance and weight which has created them. The powerful vibrations are generated via big or other neighboring spiders existing nearby the individual. The random number (r_m) is created between [0,1]. When the (r_m) value is less than the threshold value (PF) the movement of attraction is produced. The movement of repulsion is generated when the (r_m) value is less than the threshold value (PF), as given by the equation below [6]:

Received 26 Sep 2018; Accepted 4 Dec 2018

$$f_i^{t+1} = \begin{cases} f_i^t + \propto.\mathrm{Vibci}.\left(s_c - f_i^t\right) + \beta.\mathrm{Vibbi}.\left(s_b - f_i^t\right) \\ +\delta.(\mathrm{rand} - 0.5),\ \text{with probability (PF)} \\ f_i^t - \alpha.\mathrm{Vibci}.\left(s_c - f_i^t\right) - \beta.\mathrm{vibbi}.\left(s_b - f_i^t\right) \\ +\delta.(\mathrm{rand} - 0.5), \text{with probability } (1 - \mathrm{PF}) \end{cases} \tag{20}$$

, where (t) is the iteration number and (α, β, δ) are random constants created with the range $[0,1]$. In male cooperative process, the spiders of male are classified into two types: dominant spiders (D) and non-dominant spiders (ND) with respect to their positions with reference to the median member. In the communal web, the dominant males are pulled into the closest female spider. If the weight value of the male members is larger than the median value of the male population, then the male members are called dominant spiders (D). Otherwise, the male members are called non dominant spiders (ND), when the weight value of the other males members is lower than the median value of the male population, therefore; it is considered as the median male member and it has the weight (wN_{f+m}). Next, changing positions of the male spider (N_m) can be defined as shown in eq. (21) [6].

$$m_i^{t+1} = \begin{cases} m_i^t + \alpha.Vibfi.\left(s_c - m_i^t\right) + \delta.(rand - 0.5), \\ \quad if\ wN_{f+i} > wN_{f+m} \\ m_i^t - \alpha.\left(\frac{\sum_{h=1}^{N_m} m_h^t.wN_{f+h}}{\sum_{h=1}^{N_m} wN_{f+h}} - m_i^t\right), \\ \quad if\ wN_{f+i} \leq wN_{f+m} \end{cases} \tag{21}$$

Where $\left(\frac{\sum_{h=1}^{N_m} m_h^t.wN_{f+h}}{\sum_{h=1}^{N_m} wN_{f+h}}\right)$ represents the weighted mean of the male population (M). Finally, in the mating process, the female members and dominant males execute mating in social spider colony. Thus, an overwhelming male (m_g) spider finds a set (E^g) of female individuals inside a specific extend (r), which mates and shapes a new offspring (s_{new}). It is significant to increase that if the set (E^g) is unfilled, the mating task is dropped. The range (r) is defined as a radius which relies upon the size of the inquiry space, which is computed as follows (22) [6].

$$r = \frac{\sum_{j=1}^{n}(p_j^{high} - p_j^{low})}{2*n} \tag{22}$$

At last, the mating procedure is done. The spiders with heavier weight will probably impact the new offspring. The impact likelihood (p_{si}) of every part is allotted by the roulette wheel method, which is characterized as shown in equation in eq. (23) [6]:

$$p_{si} = \frac{w_i}{\sum_{j \in T^g} w_j} \tag{23}$$

Where (T^g) are all the elements generated by $(E^g \cup m_g)$ and $(\mathrm{i} \in T^g)$. The flowchart of the SSO algorithm is described in (Fig. 2).

IV. DESIGN AND SIMULATION FOR THE PROPOSED CONTROLLER

The proposed SNN controller, shown in Fig. 3, has three neurons in the input layer (H), and one neuron in the output layer (J). Each connection between neurons consists of certain number of synapses. The first input of the controller represents the error, the second input is integral of error and the third one is derivative of error. The controller uses two selection lines (c1 and c2) to select the type of the controller (see Table 1). Moreover, there are three selection lines (h1, h2 and h3) to choose the number of the neurons in the hidden layer (I) and three selection lines (s1, s2 and s3) to select the number of synapses in each sub connection (see Table 2). The controller is designed in discrete form with different sampling times according to the model to be controlled. The number of hidden nodes,

Received 26 Sep 2018; Accepted 4 Dec 2018

number of synapses, decay time constant (τ), threshold value (th), maximum time interval (T_max) and minimum time interval (T_min) are selected using trial and error to obtain the best output time response.

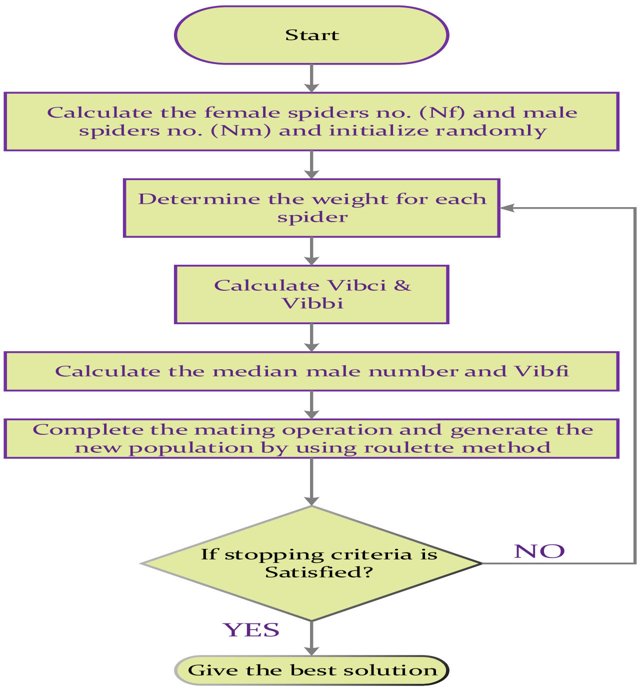

Start
Calculate the female spiders no. (Nf) and male spiders no. (Nm) and initialize randomly
Determine the weight for each spider
Calculate Vibci & Vibbi
Calculate the median male number and Vibfi
Complete the mating operation and generate the new population by using roulette method
If stopping criteria is Satisfied?
NO
YES
Give the best solution

FIG. 2: FLOWCHART OF THE SSO ALGORITHM.

In this paper, linear and nonlinear models are connected with the P/PI/PD/PID-like SNN controller, (see Fig. 4). The type of the controller is specified according to the required output response of the controlled system. Set-point input is applied with applying sudden loads. The parameters of SNN and the SSO algorithm are initialized. The weights of the hidden and output layers for SNN are trained using SSO algorithm. The weights for each spider (weights of SNN) are set in the SNN controlled system and the MSE (see equation 8) is calculated. The training algorithm stops when the best weights are obtained. That is, reaching the minimum MSE with minimum number of iterations. Finally, the best weights are set in the SNN and the controller is tested under no load and load conditions.

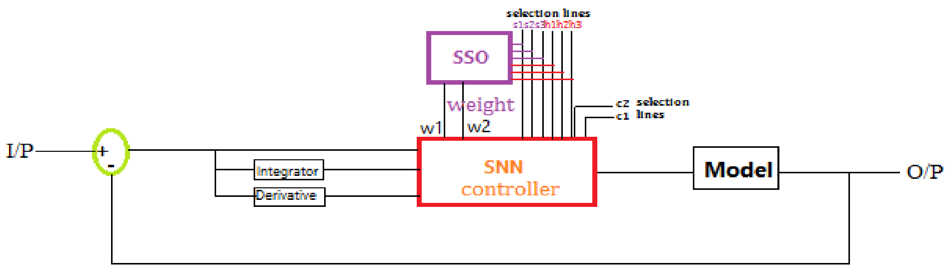


FIG. 3: BLOCK DIAGRAM OF THE PROPOSED SNN CONTROLLED SYSTEM.

Three models (linear and nonlinear) are taken to test the efficiency of the proposed controller for variable set-point inputs and different structure controllers. For the two linear models below, the following SNN parameters are selected: τ= 6msec, th= 0.8, T_{max}= 20msec and T_{min}=1 msec. The delay interval of the connection is 3 msec. Thus; the synaptic delays are from 1 to 4 msec. For SSO algorithm, the following parameters are selected: Ns= 25 spiders, the upper and lower initial weights are within (-10,10). After

Received 26 Sep 2018; Accepted 4 Dec 2018

several trials for all the tested models, it is found that the best values for the parameters of SNN are six neurons in hidden layer (I) with four synapses for each node. As a result, there are 96 weights trained in each iteration to give the best weights.

TABLE 1: SELECTION LINES FOR THE TYPE OF CONTROLLER.

Control signals C2 C1	Controller type
0 0	P-Like SNN
0 1	PD-Like SNN
1 0	PI-Like SNN
1 1	PID-Like SNN

TABLE 2: SELECTION LINES FOR THE NUMBER OF SYNAPSES AND HIDDEN NODES USED IN SNN.

Selection lines			No. of synapses (S) / No. of hidden nodes (h)
S3 h3	S2 h2	S1 h1	
0	0	1	One
0	1	0	Two
0	1	1	Three
1	0	0	Four
1	0	1	Five
1	1	0	Six
1	1	1	Seven

The linear and nonlinear models that are simulated are shown below:

1) DC Motor Linear First Order Model:

The speed control of DC motor is vastly used in control systems, like electric vehicles, machines, robots etc. The high tracking behavior, no overshoot and no oscillation are the main properties required for DC motor systems. The transfer function of the speed control of DC motor is [18]:

$$G_{(s)} = \frac{K_t}{JS+B} \tag{24}$$

, where (J) is the moment of inertia, (B) is the damping coefficient and (K_t) is the torque constant. The nominal parameters of the model with the external load disturbance (Td) equals to (0 Nm) are [18]:

$$J^* = 5.77\times 10^{-2}\, Nms^2$$
$$B^* = 8.8\times 10^{-3} Nms/rad$$
$$K_t^* = 0.667 Nm/A$$

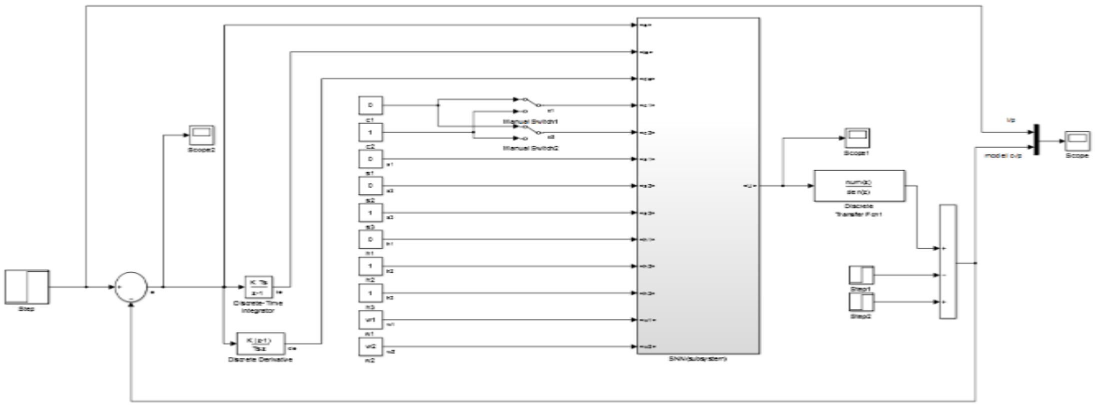


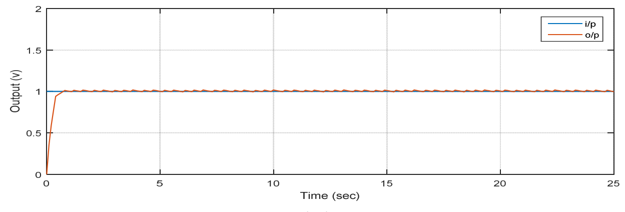
FIG. 4: SIMULINK STRUCTURE OF P/PI/PD/PID-LIKE SNN CONTROLLER.

Received 26 Sep 2018; Accepted 4 Dec 2018

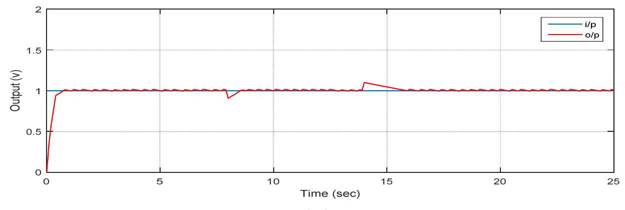
By taking J=3*J^* , B=B^* and $K^t = K_t^*$, with a sampling time TS= 0.1sec, the transfer function in discrete form is:

$$G_{(z)} = \frac{0.3843}{z-0.9949} \tag{25}$$

The closed loop controlled system is trained to work as a P-like SNN controller. Afterword, the closed loop controlled system is tested by a unit step input applied without applying a load to the system using a P-like SNN controller. The step response is shown in Fig. (5-A). Moreover, the same reference input is applied with applying a sudden load of 10% of the reference input at the time of 8 sec and removed at the time of 14 sec, see Fig. (5-B). In this system, the MSE equals to (1) after 50 iterations.



(A)



(B)

FIG. 5: SIMULATION OF P-LIKE SNN FOR A DC MOTOR FIRST ORDER LINEAR MODEL: A) WITHOUT LOAD (B) WITH LOAD.

2) Second Order Linear Model:

The following 2nd order linear transfer function is used as another model to simulate and test the proposed controller:

$$G_{(s)} = \frac{2}{s^2+4*s+3} \tag{26}$$

By using a sampling time TS=0.05sec, the transfer function in discrete form is:

$$G_{(z)} = \frac{0.00234z+0.002189}{z^2-1.812*z+0.8187} \tag{27}$$

The closed loop controlled system is trained to work as a PI-like SNN controller. A sine wave reference input, $[0.5 + 0.5 \sin wt]$ is applied with w=1 rad/sec. A sudden load equals to (10%) is applied at the time of 5 sec and removed at the time of 11sec. The SNN controller reached the best weights with MSE=2 after 80 iteration (see Fig. 7).

In addition, the closed loop controlled system is tested by applying a unit step input with the effect of applying sudden load of 10% of the reference input at the time of 5 sec and it was removed at the time of 12sec. The output response is shown in Fig. 8.

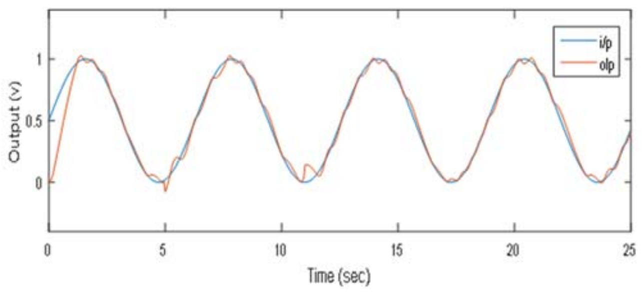


FIG. 7: SIMULATION OF PI-LIKE SNN CONTROLLER FOR A SECOND ORDER MODEL USING SINE WAVE INPUT AND A LOAD OF (10%).

Received 26 Sep 2018; Accepted 4 Dec 2018

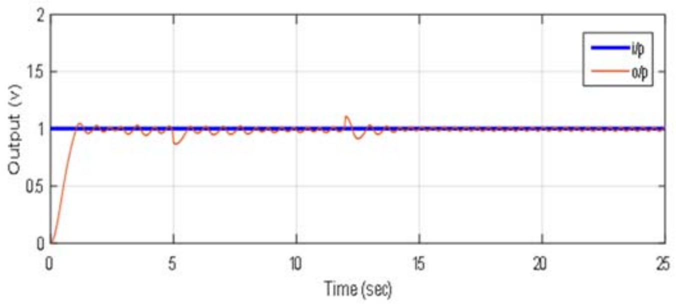


FIG. 8: SIMULATION OF PI-LIKE SNN CONTROLLER FOR A SECOND ORDER MODEL USING UNIT STEP INPUT AND A LOAD OF (10%).

3) Nonlinear servo motor:

The mathematical model of the servo motor system is represented by the second order dynamic system with friction [19]:

$$J = u - F - TL \tag{28}$$

Where (J) is the moment of inertia, $(\ddot{X})$ is the servo motor acceleration, (u) is the control input torque, (F) is the friction torque and (TL) is the load torque. The friction torque is consisted of coulomb friction (F_c), stiction friction (F_c), and the viscous friction (σ) [19] :

$$F = \left\{ F_s e^{-(\frac{\dot{X}}{\dot{X}_s})2} + F_c\left(1 - e^{-\left(\frac{\dot{X}}{\dot{X}_s}\right)2}\right) + \sigma\left|\dot{X}\right| \right\} * sgn(\dot{X}) \tag{29}$$

Where $\dot{X}_s$ is the stribeck velocity. By substituting these two equations: $e_1 = X - X_d$ and $e_2 = \dot{X} - \dot{X}_d$ into eq. (28), the following state-space equations are obtained [19]:

$$\dot{e}_1 = e_2 \tag{30}$$

$$\dot{e}_2 = \left(\frac{1}{J}\right)(u - F - TL) - \ddot{X}_d \tag{31}$$

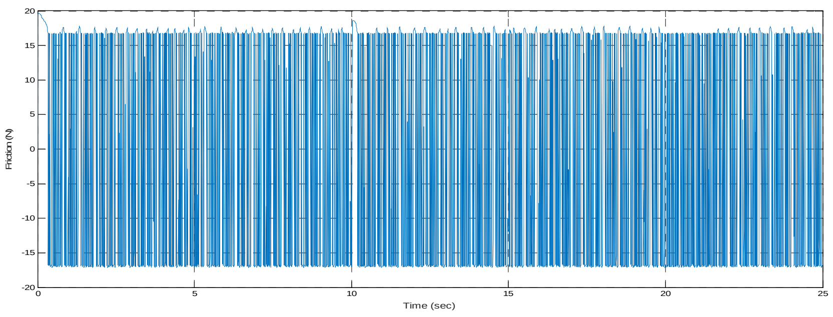
Where X_d, $\dot{X}_d$, and $\ddot{X}_d$ are the desired position, velocity and acceleration, respectively. The nominal parameters for the system that will be used in the simulation are given in Table 3 [19] with a sampling time TS=0.01sec and TL=2 N. For this model, the following SNN parameters are selected: τ= 7msec, th= 0.8, T_{max}= 15msec and T_{min}=1 msec. The delay interval of the connection is 5 msec. Thus; the synaptic delays are from 1 to 6 msec. For SSO algorithm, the following parameters are selected: Ns= 37 spiders, the upper and lower initial weights are within (-50, 50). After several trials, it is found that the best values for the parameters of SNN are nine neurons in hidden layer with six synapses for each node.

TABLE 3: THE NOMINAL PARAMETERS VALUE OF SERVO MOTOR.

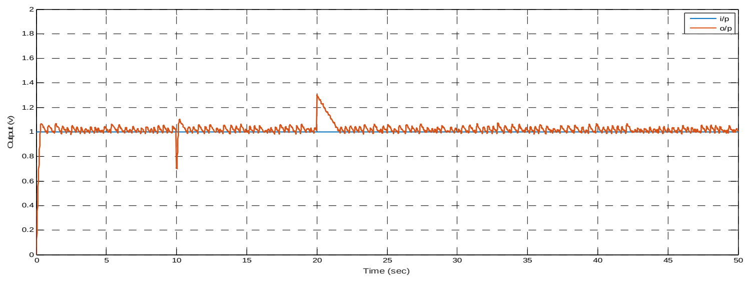
Nominal Parameters	Definition	Value	Unit
J°	Moment of inertia	0.2	Kgm^2
F_s°	Stiction friction	2.190	Nm
F_C°	Coulomb friction	16.690	Nm
$\dot{X}_s^\circ$	Stribeck velocity	0.010	rad/sec
σ°	Viscous friction	0.650	Nm.sec/rad

As a result, there are 216 weights which are trained in each iteration to give the best weights. The closed loop controlled system is trained to work as a PID-like SNN controller by applying a unit step input and a load of 30% of the reference input at the time of 10 sec and removing it at the time of 20sec. The friction applied in this system has a high nonlinearity, see Fig. (9-a). The MSE reaches the minimum value of (5) at 150 iterations and the output response is shown in Fig. (9-b).

Received 26 Sep 2018; Accepted 4 Dec 2018



(A)



(B)

FIG. 9: (A) APPLIED FRICTION OF NONLINEAR SYSTEM WITH HIGH NONLINEARITY. (B) SIMULATION OF PID-LIKE SNN CONTROLLER FOR NONLINEAR SYSTEM WITH LOAD OF (30%).

V. CONCLUTIONS

In this paper, the SNN controller is designed to work with different structures; as a P, PI, PD or PID like. It is tested on the linear and nonlinear models using different reference input signals under no load and sudden load conditions. SNN has the advantage of using smaller network size and as a result fast feed-forward calculations as compared with second generation neural networks. The SSO algorithm succeeded to converge and obtain the best weights of the SNN for different types of models and different types of reference inputs in a minimum number of iterations and more speed to reach the target goal as compared with different training algorithms for the SNN. Moreover, the proposed controller has an advantage of a free choice for the type of the desired controller.

REFERENCES

[1] D. Floreano, N. Schoeni, G. Caprari and J. Blynel, "Evolutionary Bits ’n’ Spikes", Proceedings of the Eighth International Conference on Artificial Life, 2002.

[2] W. Maass, "Networks of Spiking Neurons the Third Generation of Neural Network Models", Neural Networks, vol. 10, no. 9, pp. 1659-1671, 1997.

[3] S. Silva and A. E. Ruano, "Application of the Levenberg-Marquardt Method to the Training of Spiking Neural Networks", In Proceedings of the 2006 IEEE International Joint Conference on Neural Networks, Vancouver, BC, Canada, pp. 3978-3982, Jul. 2007.

[4] A. Kasinski and F. Ponulak, "Comparison of Supervised Learning Methods for Spike Time Coding in Spiking Neural Networks", International Journal of Applied Mathematics and Computer Science, vol. 16, no. 1, pp.101-113, 2006

[5] S. McKennoch, D. Liu, and L. G. Bushnell, "Fast Modifications of the SpikeProp Algorithm", In Proceedings of the 2006 IEEE International Joint Conference on Neural Networks, Vancouver, BC, Canada, pp. 3970-3977, Jul. 2007.

[6] H. M. Zawbaa, E. Emary, A. E. Hassanien, and B. Parv, "A Wrapper Approach for Feature Selection Based on Swarm Optimization Algorithm Inspired in the Behavior of the Social-Spider", 7th International Conference on Soft Computing and Pattern Recognition, pp. 25-30, 2015.

[7] Z. Cao, L. Cheng, and C. Zhou, "Spiking Neural Network Based Target Tracking Control for Autonomous Mobile Robots", Neural Computer Applications, vol. 26, no. 8, pp. 1839–1847, 2015.

[8] T. S. Clawson, S. Ferrari, S. B. Fuller, and R J. Wood, "Spiking Neural Network (SNN) Control of a Flapping Insect-Scale Robot", Proc. IEEE Conf. Decision and Control, pp. 3381–3388, 2016.

[9] E. Hernancez, A. Espinal, P. Mendoza, C. Capulin, R. Troncoso and H. Gonzalez, "A FPGA-Based Neuromorphic Locomotion System for Multi-Legged Robots", vol. 5, pp. 8301-8312, June 7, 2017.

Received 26 Sep 2018; Accepted 4 Dec 2018

[10] A. Antonietti, C. Casellato, E. D'Angelo and A. Pedrocchi, "Bioinspired Adaptive Spiking Neural Network to Control NAO Robot in a Pavlovian Conditioning Task", 7th IEEE International Conference on Biomedical, Netherlands, pp.142-147, August, 2018.
[11] F. Xue, W. Wang, N. Li, and Y. Yang, "FPGA Implementation of Self-organized Spiking Neural Network Controller for Mobile Robots", Advances in mechanical Engineering, Hindawi Publishing Corporation, 2014.
[12] Y. Oniz, O. Kaynak, and R. H.Abiyev, "Spiking Neural Networks for the Control of a Servo System", International Conference on Mechatronics (ICM), pp. 94-98, Feb. 27-March 1, 2013.
[13] S. M. Bohte, J. N. Kok, and H. L. Poutre, "Error-backprogation in Temporally Encoded Networks of Spiking Neurons", Neurocomputing, vol. 48, no. 1-4, pp. 17-37, 2002.
[14] V. Thiruvarudchelvan, J. W. Crane, and T. Bossomaier, "Analysis of Spike Prop Convergence with Alternative Spike Response Functions", IEEE Symposium on Foundations of Computational Intelligence (FOCI), Singapore, pp. 98-105, 2013.
[15] C. E. Klein, E. H. Segundo, V. C. Mariani, L. d. S. Coelho, "Modified Social-Spider Optimization Algorithm Applied to Electromagnetic Optimization", IEEE Transactions on Magnetics, vol. 52, no. 3, pp. 1-4, 2016.
[16] A. Ewees, M. Abd El Aziz, and M. Elhoseny, "Social-Spider Optimization Algorithm for Improving ANFIS to Predict Biochar Yield", 8th International Conference on Computing, Communication and Networking Technologies (8ICCCNT), July 3-5, Delhi-India, 2017.
[17] H. Shayeghi, A Molaee, and A. Ghasemi, "Optimal Design of FOPJD Controller for LFC in an Interconnected Multi-source Power System", International Journal on Technical and Physical Problems of Engineering, vol. 8, no. 1, pp. 36-44, 2016.
[18] H. P. Pang and Q. Yang, "Optimal Sliding Mode Control for a Class of Uncertain Nonlinear Systems Based on Feedback Linearization", Robust Control, Theory and Applications, pp. 141-162, 2001.
[19] W. Xie, "Sliding Mode Observer Based Adaptive Control for Servo Actuator with Friction", IEEE Transaction on Industrial Electronics, vol. 54, no. 3, pp. 1517-1527, June 2007.

Received 26 Sep 2018; Accepted 4 Dec 2018