Self-Tuning PID Controller Using a Neural Network for Nonlinear Exoskeleton System

Mikhail P. Belov
Department of Robotics and Industrial Automation
Saint-Petersburg Electrotechnical University "LETI"
Saint Petersburg, Russia
milesa@mail.ru

Dinh Dang Truong
Department of Robotics and Industrial Automation
Saint-Petersburg Electrotechnical University "LETI"
Saint Petersburg, Russia
dinhdangtruong@gmail.com

Pham Van Tuan
Le Quy Don Technical University
Hanoi, Vietnam
tuanhvhq@gmail.com

2021 II International Conference on Neural Networks and Neurotechnologies (NeuroNT) | 978-1-6654-4534-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/NEURONT53022.2021.9472852

***Abstract*—The article proposes the application of method analog neural networks (NN) with a radial basis function (RBF) and self-tuning of the proportional-integral derivative (PID) controller coefficients for a nonlinear control system of the lower extremities of the exoskeleton. The implementation of accurate and high-quality control of nonlinear systems, including parameter uncertainties and external disturbances, is possible using an analog neuron of a network controller, which has the ability to continuously learn and adapt. The NN in the PID controller allows us to correct errors that arise due to uncertainties and changes in parameters during the movement of the lower extremities of the exoskeleton. The effect of the proposed control algorithm is demonstrated by means of simulation in the Matlab / Simulink environment.**

Keywords— neural network, exoskeleton, self-tuning, control system, nonlinear system

I. INTRODUCTION

Nowadays, the research and development of modern exoskeletons is an urgent task for scientists around the world. The structure of the exoskeleton is a system of complex electromechanical connections, in which it is impossible to fully measure all the states of the system. In other words, it is impossible to build an accurate dynamic model with all nonlinear components, taking into account the dynamic changes associated with the movement of the carrier. One of the solutions to these problems is the use of adaptive controllers, which are widely studied in [1, 2, 3]. The PID controller is widely used in various control systems due to its simple structure and implementation. One of the biggest design challenges is finding the right gain value. The use of PID controller for nonlinear systems and systems with changing parameters during operation is not effective.

NN are an effective tool for managing nonlinear systems. They have advantages such as adaptive learning, fault tolerance, and generalization. Fierro and Lewis [4] developed a regulator based on an artificial neural network that combines a method of speed control with feedback and torque control by a multilayer neural network in a forward channel. But the structure of the controller and the learning algorithm of the neural network are very complex and require large computational resources. This paper presents the possibility of using a PID controller and an NN capable of self-learning and adapting in a nonlinear control system. However, there is a linear PID controller in the literature tuned using NN, but this can usually degrade control performance [5] if there is strong non-linearity in the controlled process. Therefore, this article proposes a non-linear PID controller tuned using NN, and it can improve the control performance of non-linear systems. Building the structure of neuron adaptive PID regulator.

II. BUILDING THE STRUCTBRE OF A NEUROADAPTIVE PID REGULATOR

A. Diagram of the structure control systems

The structure of the proposed control algorithm with neural networks based on PID controllers is shown in Fig. 1, where $e_p = q_d - q$ – joint trajectory tracking error.

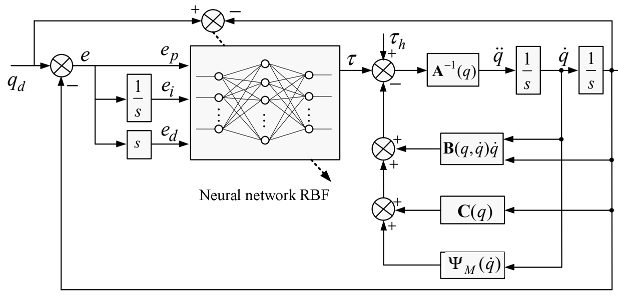

Fig. 1. The structure of the neuro adaptive PID controller for the exoskeleton

This control algorithm has a simple structure, using the method of constant auto-tuning for the NN controller, calculated time is short.

The lower limb exoskeleton is a system with multiple degrees of freedom and high-order nonlinearities, which makes it difficult to assess the accuracy of its dynamics. Mathematical model of the lower limb system of a human exoskeleton with nonlinear mechanical characteristics can be written as:

$$\mathbf{A}(q)\ddot{q} + \mathbf{B}(q,\dot{q})\dot{q} + \mathbf{C}(q) + \Psi_M(\dot{q}) = \tau + \tau_h \tag{1}$$

where:

$$\mathbf{A}(q) = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \mathbf{B}(q,\dot{q}) = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}, \mathbf{C}(q) = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix},$$

$$\tau = \begin{bmatrix} \tau_1 & \tau_2 \end{bmatrix}^T, \tau_h = \begin{bmatrix} \tau_{h1} & \tau_{h2} \end{bmatrix}^T;$$

where $q \in \mathbb{R}^n$ – generalized link position coordinate; $\mathbf{A}(q)$ – inertia matrix, is positive definite; $\mathbf{B}(q,\dot{q})$ – matrix of centrifugal and Coriolis forces; $\mathbf{C}(q)$ – vector of gravity moments; τ_h – the vector of the moment of person's leg (considered external disturbance); τ – вектор управляющего воздействия. Функция трения $\Psi_M(\dot{q})$ represented by Coulomb friction and viscous friction.

B. Neural network RBF

The RBF network is a feed-forward neural network that is trained using supervised learning techniques. Radial functions represent a special class, a characteristic feature of which is a monotonically decreasing or increasing response from a central point [6, 7]. RBF nets are capable of approximating any reasonable mapping of continuous functions with a satisfactory level of accuracy [5, 8]. A typical RBF consists of three layers: input, hidden, and output. The links between the layers are multiplied by the weights of the corresponding dimensions. The structure of the RBF neural network is shown in Fig. 2.

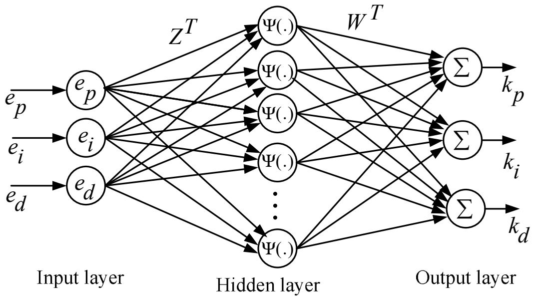

Fig. 2. The structure of the RBF neural network

where: input layer of neural network $\mathbf{x} = \begin{bmatrix} e_p & e_i & e_d \end{bmatrix}^T$; hidden layer: 10 neurons; output layer $\mathbf{Y} = \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix}^T = \begin{bmatrix} k_p & k_i & k_d \end{bmatrix}^T$. Each output of the hidden layer can be of the following form:

$$\Psi_i(x) = \exp\left(-\frac{\|x - c_i\|^2}{2b_i^2} \right), \ (i = 1, 2, ..., 10)$$

where c_i – the central position of the Gaussian function in the neurons of the hidden layer; b_i – width of the Gaussian function in the hidden layer of neurons; $\Psi(x) = [\Psi_1(x), \Psi_2(x), ..., \Psi_{10}(x)]^T$ – the output vector of the hidden layer neurons. The output layer equation can be written as:

$$y_j = \sum_{i=1}^{10} w_{ij} \Psi_i(x), \ (i = 1, 2, ..., 10; j = 1, 2, 3)$$

where w_{ij} – weights between the neurons of the hidden and output layers. The network structure with a three-layer neuron is shown in Fig. 3.

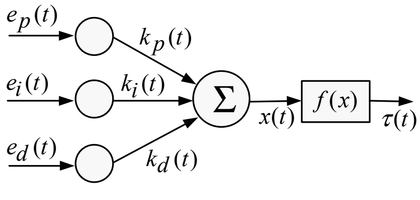

Fig. 3. Scheme of a three-layer neuron based on a PID controller

where $\tau(t)$ – output of neural network; x – sigmoid function input $f(.)$, we have nonlinear dependence in the form of a function:

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2}$$

III. Control System of the Neuron Adaptive PID Regulator

Direct control of an analogue RBF neural network based on a PID controller $\tau(t)$ is used for tracking control of the trajectory of exoskeleton joints. The regulator output can be obtained from the following equation:

$$\tau(t) = f(x) \tag{3}$$

The input to the sigmoid function in the output layer is defined as follows:

$$x(t) = k_p(t)e_p(t) + k_i(t)e_i(t) + k_d(t)e_d(t) \tag{4}$$

где $e_p(t) = q_d(t) - q(t)$, $e_i(t) = \int_0^t e_p(t)dt$, $e_d(t) = \frac{de_p(t)}{dt}$

Neural networks are trained using a conventional backpropagation algorithm to minimize the system error between the output values of the angular position of the exoskeleton and the desired one, determined from the following equation:

$$E(t) = \frac{1}{2}(e_p(t))^2 \tag{5}$$

The following equations can be obtained from the discrete algorithm using the steepest descent method [5], [9].

$$\begin{cases} k_p(t) = k_p(0) - \eta_p \int_0^t \frac{\partial E(t)}{\partial k_p} dt, \\ k_i(t) = k_i(0) - \eta_i \int_0^t \frac{\partial E(t)}{\partial k_i} dt, \\ k_d(t) = k_d(0) - \eta_d \int_0^t \frac{\partial E(t)}{\partial k_d} dt. \end{cases} \tag{6}$$

where η_p, η_i, η_d – the learning rate that determines the rate of convergence. Transforming equation (5), we get:

$$\begin{cases} \frac{\partial E(t)}{\partial k_p} = \frac{\partial E(t)}{\partial q} \frac{\partial q}{\partial \tau} \frac{\partial \tau(t)}{\partial x} \frac{\partial x}{\partial k_p} = -e_p(t) \frac{\partial q}{\partial \tau} f'(x(t)) e_p(t), \\ \frac{\partial E(t)}{\partial k_i} = \frac{\partial E(t)}{\partial q} \frac{\partial q}{\partial \tau} \frac{\partial \tau(t)}{\partial x} \frac{\partial x}{\partial k_i} = -e_p(t) \frac{\partial q}{\partial \tau} f'(x(t)) e_i(t), \\ \frac{\partial E(t)}{\partial k_d} = \frac{\partial E(t)}{\partial q} \frac{\partial q}{\partial \tau} \frac{\partial \tau(t)}{\partial x} \frac{\partial x}{\partial k_d} = -e_p(t) \frac{\partial q}{\partial \tau} f'(x(t)) e_d(t). \end{cases} \tag{7}$$

We write the derivative of equation (2) as:

$$f'(x) = \frac{3e^{-2x}}{\left(1 + e^{-2x}\right)^2} \tag{8}$$

For convenience, as in the work of Yamada and Yabut [10], we have:

$$\frac{\partial q(k)}{\partial u} = 1 \tag{9}$$

Substituting equations (8), (9) into equations (7) and then into equation (6), we obtain:

$$\begin{cases} k_p(t) = k_p(0) + \eta_p \int_0^t e_p(t) e_p(t) \dfrac{3e^{-2x}}{\left(1+e^{-2x}\right)^2} dt, \\ k_i(t) = k_i(0) + \eta_i \int_0^t e_p(t) e_i(t) \dfrac{3e^{-2x}}{\left(1+e^{-2x}\right)^2} dt, \\ k_d(t) = k_d(0) + \eta_d \int_0^t e_p(t) e_d(t) \dfrac{3e^{-2x}}{\left(1+e^{-2x}\right)^2} dt. \end{cases} \tag{10}$$

Let us illustrate the effectiveness of the proposed NN algorithm in a PID controller by simulating the exoskeleton motion control system.

IV. Simulation Results

To simulate the control system, we take: a given trajectory of the angle of the hip joint $q_{d1} = \sin(0.5t) + \sin(0.75t) + \cos(t)$, (rad) and knee joint $q_{d2} = 0.5\sin(t)$, (rad). Initial gain of PID controller $k_p = \mathbf{diag}\{500, 200\}$; $k_i = \mathbf{diag}\{80, 60\}$; $k_d = \mathbf{diag}\{30, 30\}$.

We used the Bayesian Regularization neural network training method (*trainbr*) backpropagation with weights as follows:

b1 = [-0.49950367689623154099;
0.024977842154682083214; 0.35074128346375410548;-
0.76156068934439691276;-0.079942077742934825046;
0.93674243078055230427; 0.0089281464566229318258;-
1.1498044985131152806; -0.014466071094719871321;-
0.010521772603769135798];

W1 = [-0.178849699671383644
0.0044604676959411181797
0.17231285619157155065;0.065118367254124578936
0.12371480786058125512
0.026030875702891278362;0.17867613562056222642
0.0032600399207136070839 -
0.13709048048862976232;0.21228874175274772829
0.0065332501360735673393 0.28502935587001670958;-
0.18329430994026979507 -0.0029863939843420552953 -
0.22422027779071587728;-0.22358419843172658292
0.0012204622694466852988 0.19568878492659907975;-
0.060887217578337822499 0.12261676881041208564
0.016497631517890350844;-0.22095020233071546722 -
0.0070232496025983068288 -0.29241418332370999744;-
0.043190259276334588834 0.10734357819554612989 -
0.0076347851671893802311;-0.039338988634882693374 -
0.11440115825020892582 -0.014317538911954126243];

b2 = [-0.17921821884589267393;-
0.024057544656470676581;-0.53233152658194071716];

W2 = [-1.5988517765338208232 -
0.16840665059226397893 1.1960249915047604929
1.0095810070432040817 -1.0531083665633920177 -
1.7673597189962970688 0.42097239470496095048 -
1.0823575235012232731 -0.71049679067506477104 -
0.33948567499553228277; -0.13827241270920290206
2.349495962654474468 0.045094895309028992791 -
0.18378951488454403629 0.15468703980230497086 -
0.17869340949409298624 2.5165615198939312869
0.13077189831272648113 1.5960112565758028946 -
2.0562832340260865926; 0.93259017897433738042
0.33184275736528612288 -1.2280272095753033135
1.1505931988509703334 -1.2583041515052268977
1.338470017362430875 0.28976658143182559835 -
1.2498501187930755041 -0.33416988500927807815
0.45765032062742821983];

In Fig. 4, 6 show the results of the NN with self-tuning of the coefficients of the PID controller; in Fig. 5, 7 – working out the movement of joints along predetermined trajectories with a small error under the influence of a control action on the exoskeleton (Fig. 8).

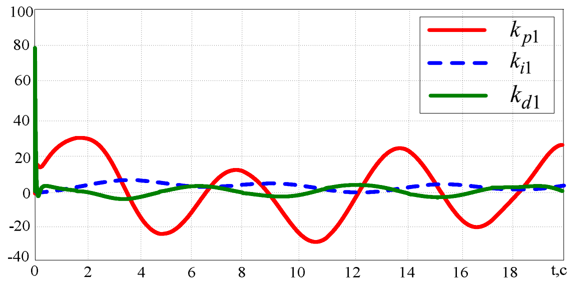

Fig. 4. The PID controller with self-tuning by NN for the hip joint

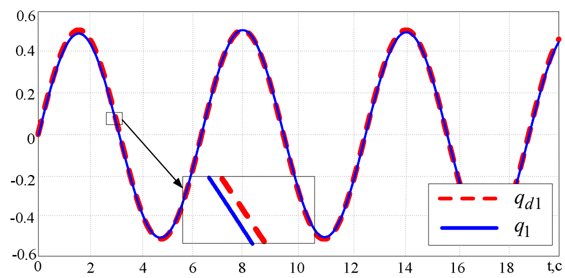

Fig. 5. Working out the movement along a given trajectory of the hip joint

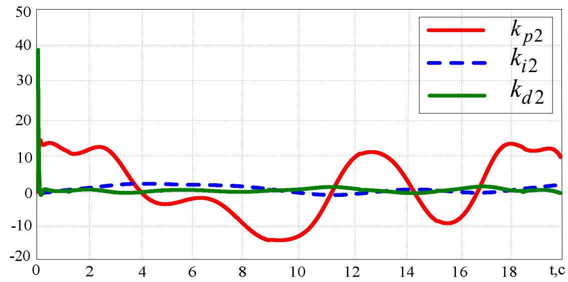

Fig. 6. The PID controller with self-tuning by NN for the knee joint

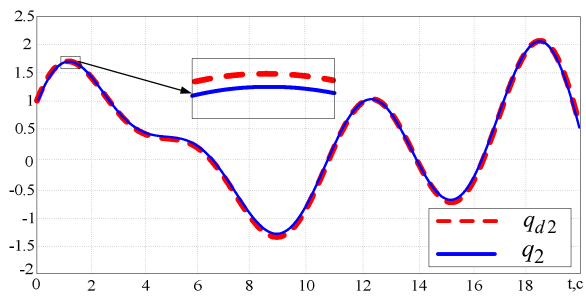

Fig. 7. Working out the movement of the trajectory of the knee joint

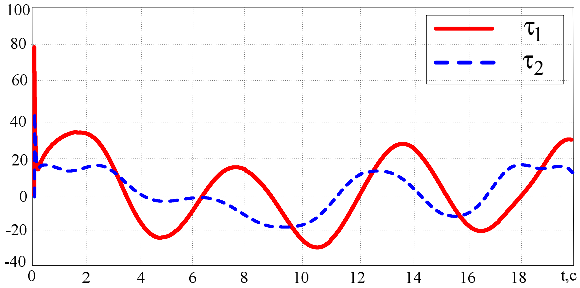

Fig. 8. Controlling effect on the exoskeleton

V. Conclusions

The presented neural controller allows for adaptive control, the optimization of the parameters is performed using the back propagation algorithm of the NN. It is resistant to disturbances, including unstructured, non-simulated dynamics. An analog NN can perform real-time control thanks to the fast online learning capability with a continuous algorithm, using only the input and output of the object to adapt the control parameters, continuously adjusting them. The results show that the proposed nonlinear PID controller with the use of NN is one of the most effective for changing the control of the exoskeleton, which is convenient for humans.

References

[1] S. L. Dai, C. Wang, M. Wang, Dynamic Learning From Adaptive Neural Network Control of aClass of Nonaffine Nonlinear Systems, IEEE Trans. on Neural Network 25 (1) (2014) 111-123.

[2] Z. Li, C. Y. Su, A. Xue, Development and Learning Control of a Human Limb With a Rehabilitation Exoskeleton, IEEE Transactions on Industrial Electronics 61 (7) (2014) 3776-3785.

[3] S. Balasubramaniana and J. Heb, Adaptive control of a wearable exoskeleton for upper-extremityneurorehabilitation, Applied Bionics and Biomechanics 9 (2012) 99-115.

[4] R. Fierro, F.L. Lewis, Control of a nonholonomic mobile robotusing neural networks, IEEE Trans. Neural Networks 9 (1998)589–600.

[5] T.D.C. Thanh, K.K. Ahn, Nonlinear PID control to improve thecontrol performance of 2 axes pneumatic artificial muscle manipulator using neural network, Mechatronics 16 (2006) 577–587.

[6] Bianchini M, Frasconi P, Gori M. Learning without local minima in radial basis function networks. IEEE TransNeural Networks1995;6:749–56.

[7] Park J, Sandberg IW. Universal approximation usingradial basis functions network. Neural Comput 1991;3:246–57.

[8] Broomhead DS, Lowe D. Multi-variable functional interpolation and adaptive networks. Complex Syst 1988;2:321–55.

[9] J. Ye, Y.P. Zhao, Application of an analog compound orthogonalneural network in robot control, in: W. Shi, S.X. Yang (Eds.),Proceedings of the International Conference on Sensing, Computingand Automation, Chonqing, China,, Watam Press, Canada, 2006,pp. 455–458.

[10] Yamada T, Yabuta T. Neural network controller using autotuning method for nonlinear functions. IEEE Trans Neural Networks 1992;3:595–601.