Optimized Neural Network Sliding Mode Control for Two Links Robot Using PSO Technique

Siham Massou, El-mahjoub Boufounas and Ismail Boumhidi

Abstract This work presents the neural network combined with the sliding mode control (NNSMC) to design a robust controller for the two-links robot system. Sliding mode control (SMC) is well known for its robustness and efficiency to deal with a wide range of control problems with nonlinear dynamics. However, for complex nonlinear systems, the uncertainties are large and produce higher amplitude of chattering due to the higher switching gain. In order to reduce this gain, neural network (NN) is used to estimate the uncertain parts of the system plant with on-line training using backpropagation (BP) algorithm. The learning rate is one of the parameters of BP algorithm which have a significant influence on results. Particle swarm optimization (PSO) algorithm with global search capabilities is used in this study to optimize this parameter in order to improve the network performance in term of the speed of convergence. The performance of the proposed approach is investigated in simulations and the control action used did not exhibit any chattering behavior.

Keywords Robot manipulators · Neural network · Sliding mode control · Particle swarm optimization

1 Introduction

The motion control design for robot manipulators attracted considerable attention, it's a complex nonlinear system and its dynamic parameters are crucial to estimate accurately. Moreover, it is almost impossible to reach exact dynamic models as the system is described by a nominal model with large uncertainties to name a few: payload parameter, internal friction, and external disturbance. To deal with the

S. Massou (✉) · E. Boufounas · I. Boumhidi
LESSI Laboratory, Department of Physics, Faculty of Sciences,
Sidi Mohammed Ben Abdellah University, Fez, Morocco
e-mail: siham.massou@gmail.com

A. El Oualkadi et al. (eds.), *Proceedings of the Mediterranean Conference on Information & Communication Technologies 2015*, Lecture Notes in Electrical Engineering 381, DOI 10.1007/978-3-319-30298-0_6

parameters uncertainties, several methods have been proposed; introducing the neural network based controls [1–4] and the Sliding Mode Control (SMC) [5, 6].

SMC is one of the most important approaches to handle systems with uncertainties, nonlinearities, and bounded external disturbance. However, there is undesirable chattering in the control effort and bounds on the uncertainties are required in the design of the SMC. It is well known that the main advantage of using the boundary layer solution [5, 7] is eliminating this chattering problem. This method can resolve the problem for only systems with small uncertainties. In case of large uncertainties, a neural network structure is proposed to estimate the unknown parts of the two-links robot model, so that the system uncertainties can be kept small and hence enable a lower switching gain to be used. The neural network weights are trained on-line using the backpropagation algorithm (BP) [8]. The proposed control consists of the predicted equivalent control added to the robust control term, where the neural network estimated function is incorporated in the equivalent control component. The learning rate is one of the parameters of BP algorithm which have a significant influence on results; practically, its value is chosen usually between 0.1 and 1 [9]. Learning rate which is too small or too large may not be favourable for convergence. In order to solve this problem, we proceed to use particle swarm optimization (PSO) algorithm with global search capabilities to optimize this parameter in order to improve the training speed.

This study is organized as follows. The next section presents the proposed optimal neural network sliding mode control. In Sect. 3, simulation results are provided to demonstrate the robust control performance of the proposed approach. Finally, in Sect. 4 a concluding remark is given.

2 Optimal Neural Network Sliding Mode Control Design

2.1 Controller Design

Consider the dynamic model of the two links robot written in the state space as follows:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = h_{1n}(\underline{x}, \underline{u}) + \xi_1(\underline{x}, t) \\ \dot{x}_4 = x_5 \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = h_{2n}(\underline{x}, \underline{u}) + \xi_2(\underline{x}, t) \end{cases} \tag{1}$$

where $h_{1n}(\underline{x}, \underline{u})$ and $h_{2n}(\underline{x}, \underline{u})$ are the nominal representations of the system, and the unknown parts $\xi_1(\underline{x}, t)$ and $\xi_2(\underline{x}, t)$. $\underline{u} = [\, u_1 \quad u_2 \,]^T$ and

$\underline{x} = [x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6]^T$. are respectively the vector inputs and the outputs of the system. More physical representation details and the lyapunov function are given in [4].

The robot manipulator control law is given as [4]:

$$\underline{u} = g_n^{-1}(\underline{x})\left(-\left(f_n(\underline{x}) + \hat{\xi}(\underline{x}, t)\right) + \begin{pmatrix} \dot{x}_{3d} \\ \dot{x}_{6d} \end{pmatrix} - \gamma\, \ddot{e} - \beta \dot{e} - ksat(S)\right) \tag{2}$$

2.2 Neural Network Representation

In this paper, we consider a NN with two layers of adjustable weights. $\underline{x}$ is the state input variables and the output variables are:$y_1 = \hat{\xi}_1(\underline{x}, t)$ *and* $y_2 = \hat{\xi}_2(\underline{x}, t)$ $y_k = W_k^T$ $\sigma(W_j^T\, \underline{x})$ $k = 1, 2$. Where $\sigma(.)$ represents the hidden-layer activation function considered as a sigmoid function given by: $\sigma(s) = \frac{1}{1+e^{-s}}$ $W_k = [W_{k1} W_{k2} \ldots$ $W_{kN}]^T$ and $W_j = [W_{j1} W_{j2} \ldots W_{jN}]^T$ are respectively interconnection weights between the hidden and the output layers and between the input and the hidden layers. The actual output $y_{dk}(\underline{x})$ (desired output which is the difference between the actual and nominal functions) is:

$$y_{dk}(\underline{x}) = y_k(\underline{x}) + \varepsilon(\underline{x}) \tag{3}$$

where $\varepsilon(\underline{x})$ is the NN approximation error.

The network weights are adjusted during the online implementation. The method used is based on the gradient descent method (GD). The essence of the GD consists of iteratively adjusting the weights in the direction opposite to the gradient of E, so as to reduce the discrepancy according to:

$$\frac{\partial W_{kj}}{\partial t} = -\eta_k \frac{\partial E}{\partial W_{kj}} \tag{4}$$

where $\eta_k > 0$ is the usual learning rate. The gradient terms $\frac{\partial E}{\partial w_{kj}}$ can be derived using the backpropagation algorithm [8]. The cost function E is defined is the error index and the least square error criterion is often chosen as follows: $E = \frac{1}{2}\sum_{k=1}^{2} \varepsilon_k^2$

2.3 PSO-Based Training Algorithm

Particle swarm optimization (PSO) is a type of derivative free evolutionary search algorithms resulted by an intelligent and well organized interaction between

individual members in a group of birds or fish for example. This algorithm was planned to simulate the positioning and dynamic movements in biological swarms as they look for sources of food or keep away from adversaries [10].

In PSO, m particles fly through an n-dimensional search space. For each particle i, there are two vectors: the velocity vector $V_i = (v_{i1}, v_{i2}, \ldots, v_{in})$ and the position vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$. Similar to bird socking and fishes schooling, the particles are updated according to their previous best position $P_i = (p_{i1}, p_{i2}, \ldots, p_{in})$ and the whole swarm's previous best position $P_g = (p_{g1}, p_{g2}, \ldots, p_{gn})$. This means that particle i adjust its velocity V_i and position X_i in each generation according to the equations bellow [11]:

$$v_{id}(t+1) = v_{id}(t) + c_1 \times rand()_1 \times (p_{id} - x_{id}) + c_2 \times rand()_2 \times (p_{gd} - x_{id}) \quad (5)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (6)$$

where $d = 1, 2, \ldots, n$; c_1, c_2 are the acceleration coefficients with positive values; $rand()_1$, $rand()_2$ are random numbers between 0 and 1. The new velocity and position for each particle are calculated using the Eqs. (5) and (6) based on its velocity $v_{id}(t)$, best position P_{id} and the swarm's best position P_{gd}.

In order to calculate the optimized parameter of learning rate η_k given in Eq. (4), the PSO is used *off-line* to minimize the neural network prediction error.

We define the quadratic errors e_{rq} as: $e_{rq}(t_i) = \frac{1}{2}\sum_{k=1}^{2} \varepsilon_k^2(t_i)$

The objective function f to be minimized is chosen as the norm of the quadratic error: $f = norm(E_{rq})$ with E_{rq} the vector that contains all errors $e_{rq}(t_i)$.

The PSO algorithm test the search space using m particles according to (5) and (6). Each particle i moves in search space and stores its best position p_{id} (η_k), then, it compares all positions to finally take out the chosen $\eta_{k-optimum}$ that give the minimum value of the objective function f.

3 Simulation Results

In this section, we test the proposed control approach on a two links robot described by the model (1). The control objective is to maintain the system to track the desired angle trajectory:$x_{1d} = (\pi/3)\cos(t)$ and $x_{4d} = \pi/2 + (\pi/3)\sin(t)$.

The masses are considered to be $m_1 = 0.6$ and $m_2 = 0.4$. The considered uncertainties are a vector random noise with the magnitude equal to unity.

$$E = \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix},\ B = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix} \text{ and } J = \begin{pmatrix} 100 & 0 \\ 0 & 100 \end{pmatrix}$$

The switching functions coefficients are defined as: $\gamma_{11} = \gamma_{22} = \beta_{11} = \beta_{22} = 4$. A swarm population of 20 particles is used in this paper (Table 1).

From Figs. 1 and 3, it can be seen that the tracking performance is obtained without any oscillatory behaviour even in the presence of large uncertainties. The corresponding control current signals are given in Figs. 2 and 4.

Table 1 The global optimum particle of learning rate η_k

Iteration number	Variation of η_k	Iteration number	Variation of η_k
1	0.4232	21	0.1124
2	0.6444	22	0.1210
3	0.8449	23	0.1191
4	0.5226	24	0.1075
5	0.6521	25	0.1024
6	0.6984	26	0.1008
7	0.5981	27	0.1015
8	0.6954	28	0.1085
9	0.3215	29	0.1112
10	0.1201	30	0.1102
11	0.0544	31	0.1094
12	0.1410	32	0.1084
13	0.1727	33	0.1081
14	0.1710	34	**0.1082**
15	0.1654	35	**0.1082**
16	0.1591	36	**0.1082**
17	0.1053	37	**0.1082**
18	0.1722	38	**0.1082**
19	0.1834	39	**0.1082**
20	0.1949	40	**0.1082**

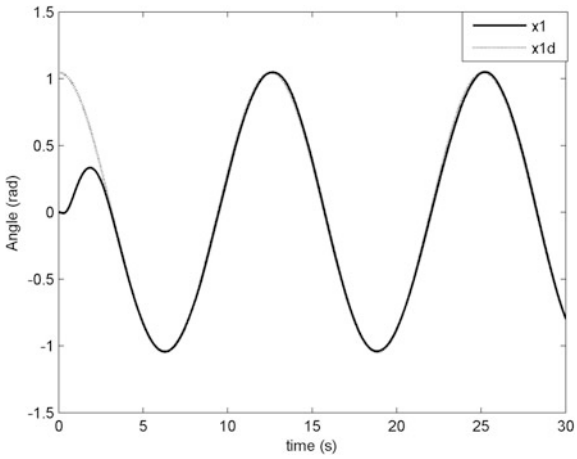

Fig. 1 Angle response x_1 and desired trajectory x_{1d}

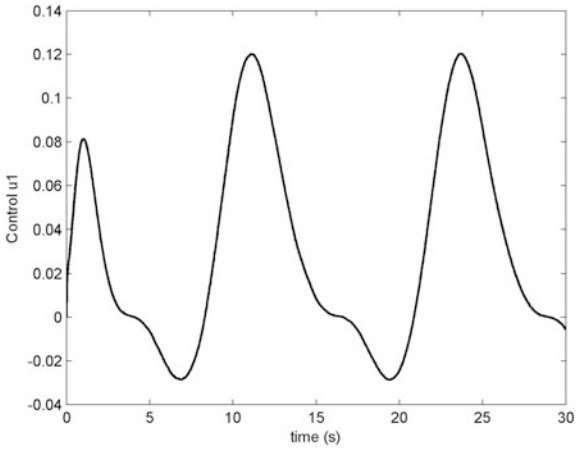

Fig. 2 Control u_1 (input current of join actuator 1)

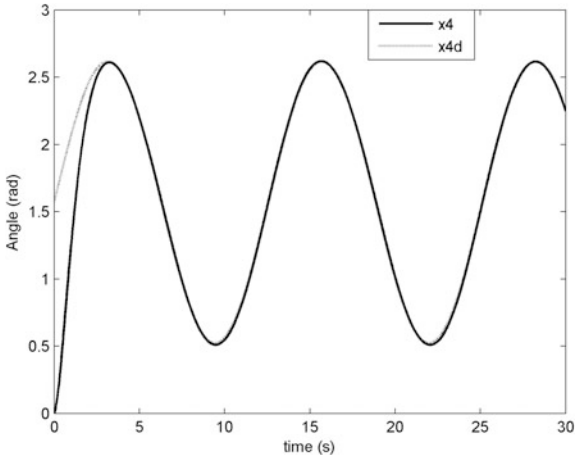

Fig. 3 Angle response x_4 and desired trajectory x_{4d}

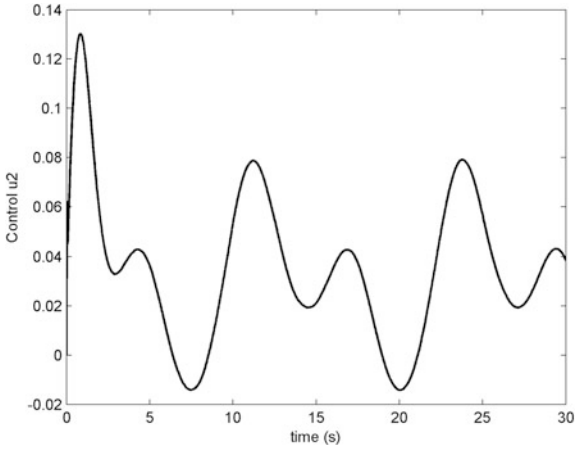

Fig. 4 Control u_2 (input current of join actuator 2)

4 Conclusion

This paper addressed the robust optimal reference tracking problem for two-links robot manipulators. The designed method is a combination of traditional sliding mode control approach and neural network. The later is employed to approximate the unknown nonlinear model function with online adaptation of parameters via BP learning algorithm. This provides a better description of the plant, and hence enables a lower switching gain to be used despite the presence of large uncertainties. The particle swarm optimization (PSO) algorithm is used to optimize the learning rate of BP algorithm in order to get faster convergence. The simulation results have shown a good performance of the proposed method to track the reference without any oscillatory behavior. Further studies would focus on more effective optimization methods for the gain of the sliding additive control.

References

1. Patino, H.D., Carelli, R., Kuchen, B.R.: Neural networks for advanced control of robot manipulators. IEEE Trans. Neural Networks **13**, 343–354 (2002)
2. Hussain, M.A., Ho, P.Y.: Adaptive sliding mode control with neural network based hybrid models. J. Process Control **14**, 157–176 (2004)
3. Liu, P.X., Zuo, M.J., Meng, M.Q.H.: Using neural network function approximation for optimal design of continuous-state parallel-series systems. Comput. Oper. Res. **30**, 339–352 (2003)
4. Sefreti, S., Boumhidi, J., Naoual, R., Boumhidi, I.: Adaptive neural network sliding mode control for electrically-driven robot manipulators. Control Eng. Appl. Inform. **14**, 27–32 (2012)
5. Slotine, J.J.: Sliding controller design for non-linear systems. Int. J. Control **40**, 421–434 (1984)
6. Utkin, V.I.: Sliding modes in control optimization, Springer (1992)
7. Slotine, J.J., Sastry, S.S.: Tracking control of nonlinear systems using sliding surfaces with applications to robot manipulators. Int. J. Control **39**, 465–492 (1983)
8. Rumelhart, D.E., Hinton, G.E., Williams, RJ.: Learning internal representations by error propagation. In: Parallel Distributed Processing, vol. 1. Cambridge, MIT Press (1986)
9. Fu, L.M.: Neural Networks in Computer Intelligence. McGraw-Hill, New York (1995)
10. Eberhar, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro-Machine and Human Science, pp 39–43 (1995)
11. Cavuslua, M.A., Karakuzub, C., Karakayac, F.: Neural identification of dynamic systems on FPGA with improved PSO learning. Appl. Soft Comput. **12**, 2707–2718 (2012)