Proceedings of the 5th International Conference of Control, Dynamic Systems, and Robotics (CDSR'18)
Niagara Falls, Canada – June 7 – 9, 2018
Paper No. 140
DOI: 10.11159/cdsr18.140

New Methodology to Design Learning Control for Robots Using Adaptive Sliding Mode Control and Multi-Model Neural Networks

Meysar Zeinali, Hao Wang
School of Engineering of Laurentian University in Sudbury
Ontario, Canada
mzeinal@laurentian.ca; HWang@laurentian.ca

Abstract - In this paper, a new methodology is proposed to design a learning control for robots. Since, the advanced robots need to work in an unstructured and dynamic environment such as human environment (e.g. assistive robots). They need to learn how to interact with people and manipulate the different objects and payloads. Additionally, due to the nonlinearity, the uncertainty of the parameters, external disturbances, and time-varying effects such as tear and wear, the accurate analytical models are too complex to derive for control applications. In this paper, a learning control is developed by effectively combining the adaptive continuous sliding mode control (ACSMC) presented in [1] with multi-model neural network techniques (MMNN). The controller consists of an online adaptation mechanism and an online learning mechanism. It is shown that learning capability allows to realize the controller with less or no prior information of robot inverse dynamic model. The robustness, performance and learning capability of the control system is demonstrated and evaluated trough simulation study and experimentally, using a two-degrees of freedom robot.

Keywords**:** Robot Learning Control, Neural Network, Adaptive Sliding Mode Control.

1. Introduction

Learning control of robots refers to the procedure of acquiring a control policy for a particular task (e.g., trajectory tracking) either by trial and error or by human supervision. The main feature of learning control is that it is permitted to fail during learning process, while in adaptive control fast convergence is required without failure [2]. Development of the learning control has become a practical approach nowadays, due to the rapid progress in computing power of affordable computers, the growing body of knowledge in artificial intelligence (AI) and robotics, which allow effective combination of the methods from classical control and methods from AI. While the combination of methods from AI and classical control covers a broad range of methods (e.g., adaptive fuzzy-SMC [3-4], neuro-fuzzy and SMC [5], neural network and SMC [6], adaptive PID Neural Networks [7] etc.) in this paper, a novel combination of ASMC and multi-model neural network are considered.

Neural networks (NN) are systems that learn from the underlying relationships of data. They are organized in a way to simulate the cells of human. Due to their capabilities for adaptation, and function approximation [8], they have proven to be well suited for control applications, i.e., modelling of the controller or the systems to be controlled. Neural networks have been used for direct and inverse system identification, which are associated with control problem of a system [9]. In both tasks, learning aspect of neural network has been used to capture the system behavior.

Direct system identification is the task of approximating the behavior of a system using a neural network. If the system maps a set of input variables “I” to output variables “O”, then direct identification is conducted by a feedforward neural network whose inputs correspond to I and outputs correspond to O. For instance, the amount of torque/force applied to a robot joint has been given and the system output are displacement, velocity and acceleration. Neural network is trained to minimize error between the actual system output (q) and neural network output (q_{NN}), where q is the position, velocity and acceleration vector of robot joints. In this case the error signal used to train the NN is $E_{train} = q - q_{NN}$. The universal approximation capability of NN is used to make the NN learn a certain highly nonlinear function that represents direct dynamics or any other characteristics of the system [10]. Inverse identification is the task of learning the inverse dynamic of the robot. The inverse problem in this case consists of determining the torque/force required to produce a desired amount of displacement, velocity and acceleration. In this case, the error signal is $E_{train} = \tau - \tau_{NN}$.

The new method of combining NN with ASMC is illustrated in Figure 1. In this method, the controller is constructed by combining the adaptive continuous sliding mode control method proposed in Zeinali and Notash [1] and the multi-model feedforward NN, which is motivated by availability of the computational speed of affordable computers. The adaptive ASMC techniques are naturally attractive because of their capability to deal with uncertainties, good transient performance (i.e., small tracking error) and fast response [11]. The NN systems have been successful due the following features: i) Realization of fast decision-making and control by parallel computation; ii) Ability to adapt to time-varying uncertainties due to large number of parameters iii) Robustness with respect to the un-modelled dynamics, due to generalization of networks [12]. As shown in Figure 1, the controller consists of two components, a multi-model neural network system that learns the inverse model of the robot in the presence of the uncertainties and produces the forces/torques to be applied to the robot, given desired positions, velocities, and accelerations, and an adaptive SMC consists of on-line estimation of system dynamic and lumped time-varying uncertainties such as external disturbances using the dynamic behaviour of a sliding function, which is based on tracking error and its derivatives (for details the reader is referred to Ref. [1]). The ASMC is also responsible to stabilize the closed-loop system and to generate a correcting signal, while NN system is in learning process. The simulation and experimental results show the proposed control method is able to deal with the tracking problem with uncertain parameters and external disturbance. Adaptation capability of the ASMC and learning capability of the NN are combined to enhance the robustness of the controller, eliminate the requirement for a priori knowledge of the bounds of uncertainties, and compensate for external disturbances and un-modelled dynamics. The multi-model neural network (i.e., Figure 2) component of the controller can be trained in an off-line training session and during on-line operations. As indicated in Figure 2, in multi-model neural network system, each model is trained based on sperate data set which are collected in different scenario and based on different trajectories. In this paper, the weights of the neural network, which are the controller parameters, are updated based on an error signal such that to stablish a sliding motion (will be described later) and is as follows.

$$E_{train} = \tau - \tau_{NN} = \tau_{SMC} \tag{1}$$

The convergence of error signal to zero in equation (1) is established using Lyapunov's approach and fundamentals of sliding mode theory, which in turn, guarantees the stability of the closed-loop system. As shown in Figure 3, the parameter of the multi-model NN is updated based on perdition error (i.e., comparison of NN model output and actual joint torque of the robot) to learn the whole dynamics of the robot.

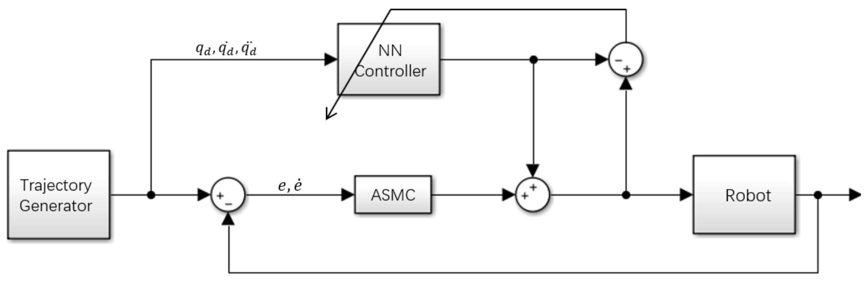


Fig. 1: Block diagram of the proposed controller.

Remark 1: The contributions of this paper are twofold: **i**) novel combination of ASMC with multi-model NN in such a way that stablishing asymptotic sliding motion is achieved through learning the whole dynamic of the robot, or regardless of the NN output, which resembles the scenario that NN is failed during leaning process. This method of combination eliminate requirement for a priori knowledge of bound of uncertainties and system dynamics; and **ii**) increasing the robustness of the controller to structured and unstructured uncertainties.

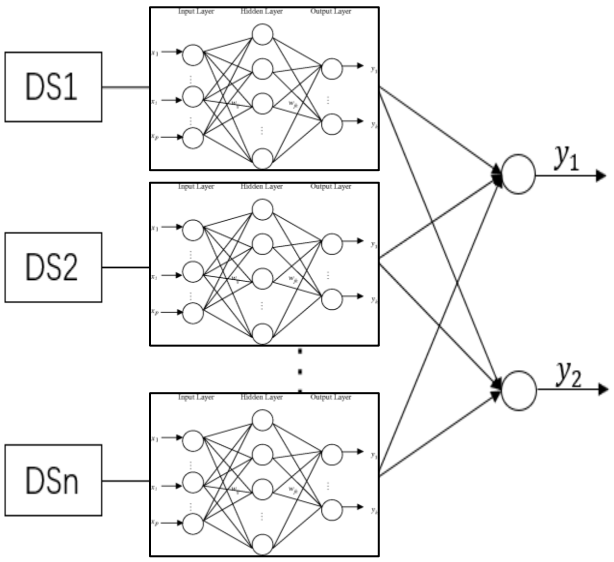


Fig. 2: Architecture of multi-model NN.

The remainder of this paper is organized as follows: Section 2 presents the system description and problem formulation. Section 3 describes the general structure of the ASMC controller. The experimental results are presented in Section 4. Section 5 concludes the paper.

2. Problem Formulation

The dynamic model of a rigid serial or parallel robot manipulator can generally be described by the following second-order differential equation, in active joint space Angeles [13].

$$\tau = M(q)\ddot{q} + h(q,\dot{q}) + d(t) \tag{2}$$

where τ is an $n\times 1$ vector of input generalized forces and n denotes the number of generalized coordinates of the manipulator. $M(q)$ is an $n\times n$ manipulator inertia matrix which is symmetric positive definite, $q \in {}^{\circ}\,{}^{n}$ is the position vector of joints, $\dot{q} \in \mathbb{R}^n$ and $\ddot{q} \in \mathbb{R}^n$ are the velocity and acceleration vectors of joints respectively, and $h(q,\dot{q}) = C(q,\dot{q})\dot{q} + F_f(q,\dot{q}) + g(q)$ where $C(q,\dot{q},t)$ is an $n\times n$ matrix of centripetal and Coriolis terms, $F_f(q,\dot{q},t)$ is an $n\times 1$ vector denoting viscous and Coulomb friction coefficients, and $g(q)$ is an $n\times 1$ vector of gravitational terms. It should be noted that for parallel manipulators, equation (2) may contain another term due to the holonomic constraints because of the closed-loops and existence of passive joints which is $F_c^T(q)\lambda_{LG}$, where λ_{LG} is the vector of Lagrange multipliers in the realm of Lagrangian dynamics. For detailed information about parallel manipulator dynamics the reader is referred to Angeles, ([13], page 473). As pointed out earlier, because of the system uncertainty and external disturbances, equation (1), which describes the dynamic model of a manipulator, is not exactly known. Therefore, the dynamic model of the manipulators is as

$$\tau = (\hat{M} + \Delta M)\ddot{q} + \hat{h} + \Delta h + T_d \tag{3}$$

where $\hat{M}(.)$ and $\hat{h}(.)$ are the known (estimated) parts of $M(.)$ and $h(.)$ respectively, $\Delta M(.)$ and $\Delta h(.)$ are the unknown parts of $M(.)$ and $h(.)$ respectively, and T_d is an $n\times 1$ bounded vector arising from the external disturbances. In this work, without loss of generality, it is assumed that the term denoting the viscous and Coulomb friction effects is completely unknown.

Therefore, in the presence of the parameter uncertainty, un-modelled dynamics and external disturbances, Eq. (3) can be written as follows

$$\tau = \hat{M}\ddot{q} + \hat{h}(q,\dot{q},t) + \delta(q,\dot{q},\ddot{q},t) \tag{4}$$

where $\delta(q,\dot{q},\ddot{q},t)$ is an unknown function that lumps together various uncertain terms (i.e., un-modelled dynamics, parameter uncertainty, external disturbances), and defined as follows

$$\delta(q,\dot{q},\ddot{q},t) = \Delta M\ddot{q} + \Delta C\dot{q} + \Delta g + F_f(q,\dot{q}) + d(t) + F_c^T(q)\lambda_{LG} \tag{5}$$

It is worth noting that inclusion of the complexities in the vector of uncertainty has two advantages: (***i***) it reduces the order of the dynamic model and the number of unknown parameters, which are needed to be estimated or identified; and (***ii***) it also simplifies the controller design to some extent. Therefore, to develop the controller and analyze the stability of closed-loop system, the nominal dynamic model of the robot is built based on the rigid body mode and all uncertain terms lumped into uncertainty term which results in the dynamic model of the robot manipulators presented in Eq. (4) and is used throughout this paper. At this point, the following assumptions are made to design the controller and analyze the stability of the closed loop system.
Assumption 1. The uncertainty vector $\delta(q,\dot{q},\ddot{q},t)$ and its partial derivatives are bounded in Euclidian norm as:

$$\|\delta(q,\dot{q},\ddot{q},t)\| \le \rho(q,\dot{q},\ddot{q},t) \tag{6}$$

where $\rho(q,\dot{q},\ddot{q},t)$ is the unknown bounding function of the uncertainties, $\|.\|$ is Euclidian norm.

3. Brief Description of ASMC Controller Design

The standard sliding mode control law for a robot manipulator system, which guarantees the stability and convergence, is a discontinuous control law in the following form:

$$\tau = \tau_{eq} - K_b \operatorname{sgn}(S) = \tau_{eq} + \tau_{sw} \tag{7}$$

where K_b is the bounds of uncertainty vector which have to be known a priori, S is the sliding surface function, and $\operatorname{sgn}(S)$ is a signum function, τ_{eq} is the model-based component, and τ_{sw} is the discontinuous term. However, the application of control law given in equation (8) can be limited mainly due to chattering and unknown bounds of the uncertainties. The above discontinuous term is the fundamental cause of chattering and is designed conservatively based on the bounds of uncertainties. Therefore, in this study, the discontinuous term $K_b \operatorname{sgn}(S)$ in equations (7) will be replaced by a continuous term which can be updated recursively in an online procedure to compensate for uncertainties. This in turn, eliminates the chattering and there is no need to have a-priori knowledge of the bound of the uncertainties. The task of the robust continuous sliding mode control design with the above mentioned characteristics and without knowing the bounds of uncertainties consists of two phases: *i*) selection/design of a sliding (switching) surface so as to achieve the desired system behavior (e.g., asymptotic stability), when restricted to the surface; and *ii*) constructing a control law to force the system state trajectory to move towards and stay on the sliding surface. For the first phase of the design, an exponentially stable error dynamic is chosen as a desired sliding surface to guarantee the convergence of tracking error to zero, while the system is in the sliding mode. According to Slotine and Li [11] sliding surface function can be defined as:

$$S(t) = (\frac{d}{dt} + \Lambda)^2 (\int e\,dt) = \dot{e} + 2\Lambda e + \Lambda^2 \int e\,dt = 0 \tag{8}$$

where Λ is an $n \times n$ symmetric positive definite diagonal constant matrix and $e = q - q_d, \dot{e} = \dot{q} - \dot{q}_d$, are the tracking error and the rate of error respectively. q_d and q are the desired and measured joint variables respectively. The integral of error is included to ensure zero offset error. The next phase is to design a control law with variable parameters such that it guaranties the existence of sliding mode, or makes the associated Lyapunov function a decreasing function of time in the presence of uncertainties. The proposed control law that guarantees achieving the above goal is as follows, for details please refer to reference [1]:

$$\tau = \hat{M}\ddot{q}_r + \hat{C}(q,\dot{q},t)\dot{q}_r + \hat{g}(q) - KS - \Gamma\int S\,dt \tag{9}$$

If it is assumed that there is no priori information of the robot dynamics, then the above controller can be written as:

$$\tau_{ASMC} = -KS - \Gamma\int S dt \tag{10}$$

Remark 2: In references [1] and [14] it is shown that a real sliding mode can be stablished with the above control law, which means $S(t) \to \varepsilon$, where ε is a very small positive scalar value and its magnitude depends on the choice of K and Γ in equation (10), and the designer of the controller can make it arbitrarily small by using the design parameters. From one hand, if $\varepsilon \to 0$, then $S(t) \to 0$ and $\tau_{ASMC} \to 0$, and this means error signal (i.e., $E_{train} = \tau - \tau_{NN} = \tau_{SMC}$) converges to zero, which means whole dynamic of the robot learned by NN system, and this in turn, slow down the learning process of the NN. Adjusting the weights of NN based on ASMC control output makes it an adaptive learning process, and ASMC output depends on the s-dynamic. Once controller starts working, NNs component starts learning the dynamic of the system, while ASMC is responsible for stability and performance of the closed-loop system. As learning continues, ASMC output decreases and NN output increases until the total desired torque for given reference trajectory is generated by NN system (see Figure 4(b)). The performance of the controller is verified by simulation and experiment. Due to lack of space the simulation results are not included.

3.1. Computing Multi-Model Neural Network Output

The architecture of the neural network is a three-layer network including one input-layer, one hidden-layer and one output layer. The input variables are position, velocity and acceleration of each joint$(q,\dot{q},\ddot{q})$, and the output of the neural network is the torque τ_N that

$$\tau_{NN} = f(q,\dot{q},\ddot{q}) \tag{11}$$

The units in each layer is fully connected, hence, the input for the hidden neurons net_i identified by the equation:

$$net_i = \sum w_{ij} I_i \tag{12}$$

where I_i denotes the input variables and w_{ij} denotes the weight. The hidden neurons output out_i can be obtain from hyperbolic tangent function:

$$out_i = f(net_i) \tag{13}$$

The output layer used the same process, but activation function changed to linear function. Therefore, the neural network output τ_{NN} is

$$\tau_{NN} = \sum w_{ij} out_i \tag{14}$$

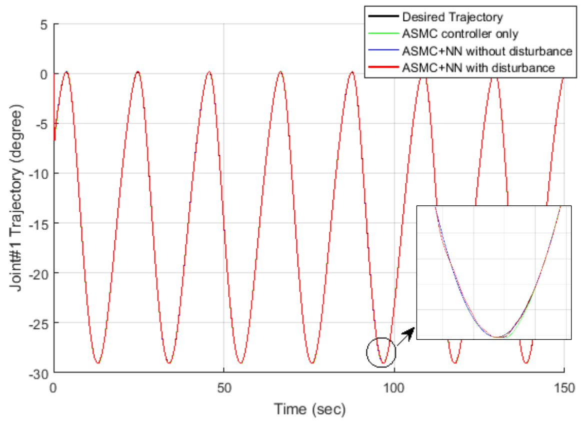
Back-propagation algorithm is used in online learning process. The weight w_{ij} can be updated by the equation:

$$w_{ij} = w_{ij} - \eta \Delta w_{ij} \tag{15}$$

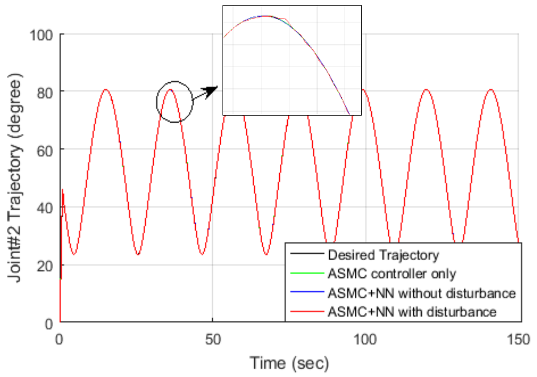
where η is a small positive number terms as learning rate. To generate a multi-model neural networks (MMNN) and to calculate the output of the, first n set of data are generated to build n NNs model which are called primary neural network model (PNNM$_1$, …, PNNM$_n$) each with three layers. Then parameters of the PNNMs (i.e., weight of the connections of each model) were tuned using back-propagation method, simultaneously the parameters of the MMNN (i.e., connection weights of the PFMs to MMNN) also adjusted to further refine the final the overall model output. The results for two-link robot data and 5 set of noisy test data are shown in Figs. 3 to 7.

4. Experimental Result

To evaluate the performance and learning capability of the proposed controller, it is implemented and tested using a 2-DOF serial flexible link Quanser robot. For experimental study, three scenarios are considered and compared as follows: 1) Trajectory tracking using ASMC controller only; 2 and 3) Trajectory tracking using multi-model NN and ASMC controller without external disturbances and with external disturbances, respectively. The desired trajectories used in the experiments are circular trajectory with diameter of 7 (cm) in Cartesian space and different angular velocities, hence, the trajectory for each joint is sinusoidal function as expressed in Fig. 3(a) and (b). The external disturbance is applied to the joint 1 and 2 at t =91(Sec) t = 35(Sec), respectively. Figs. 3(a) and (3(b) show that the high-performance tracking is achieved for all three scenarios. It is also shown that in the presence of external disturbance the actual trajectory is rapidly converging to desired one.



(a)



(b)

Fig. 3: a) Joint 1; b) Joint 2 Trajectory Tracking.

Figs. 4(a) and 4(b) explain the situation that a large external disturbance is applied to the robot. The external disturbance is applied by holding link 2 at t=44 (Sec) and releasing after about 2 seconds. The actual trajectory produces an obvious offset that is shown in Figs. 4(a) and 4(b)describe the ASMC output torque state, ASMC export torque with -20.6 (N.m) during holding and jump to 8 N.m after release then converge to zero again in 6 seconds. It illustrates that ASMC controller can maintain system stable under external disturbance.

Figs. 5(a) and 5(b) illustrate the trajectory tracking error for joint 1 and joint 2, respectively. The error jumps at the turning point due to the Coulomb friction of robot arm. If only using ASMC controller, the peak value of tracking error is around ±0.0015 degree, for joint 2, the tracking error close to ±0.002 degree. After combining with multi-model NN, the tracking error of joint 1 decreases to 0.0008 and -0.001 degrees, and joint 2 errors comes to 0.0018 and 0.001 degree.

Figs. 6(a) and 6(b) shoes the neural network output of each joint for the cases with and without disturbance. The output torque increases to a high value when the disturbance is applied to the joint which is an indication of the learning capability of the NN. It also shows that the disturbance act on the joint 2 has significant influences on the joint 1 multi-model NN output, however, the disturbance applied to the joint 1 has much less affect to the joint 2 neural network output. Figs. 7(a) and 7(b) elaborate the corresponding neural network training error, the performance expresses by root mean square error. Without disturbance, the RMS of joint 1 multi-model NN converges to 0.039, and joint 2 multi-model NN RMS converges to 0.085. Applying disturbance to the joint increases the RMS value, for joint 1 RMS is 0.044, and for joint 2 it increases to 0.094.

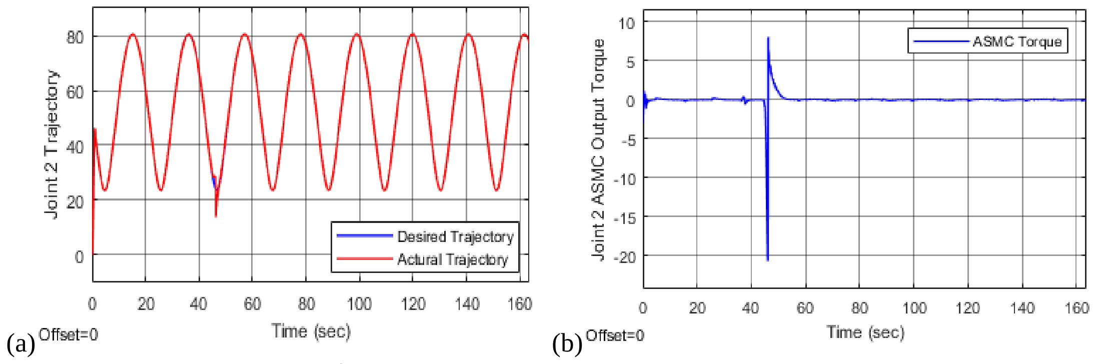


Fig. 4: a) Trajectory tracking in the presence of disturbance; b) $\tau_{ASMC} \to 0$ as described in Section 3 and compensates for the external disturbance torque very rapidly.

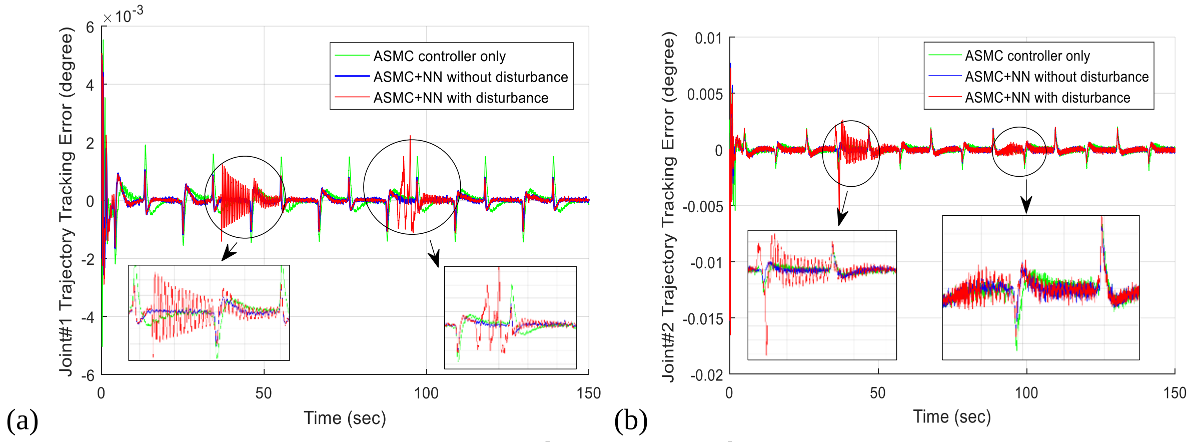


Fig. 5: Joint 1 and 2 trajectory tracking error.

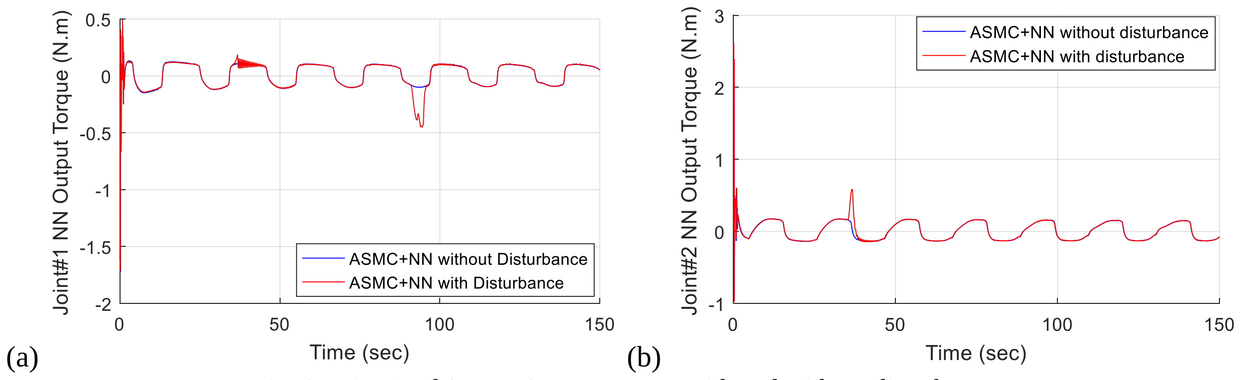
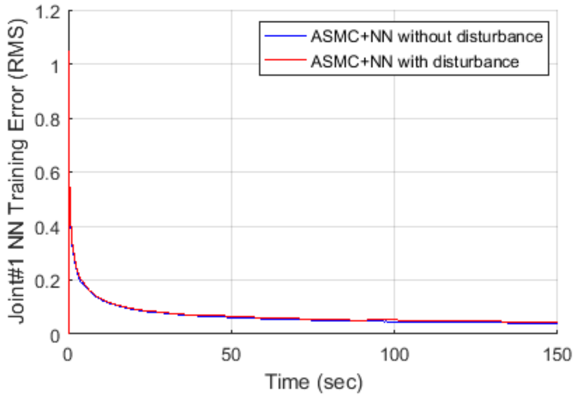


Fig. 6: Joint 1and 2 NNs Output Torque with and without disturbance.



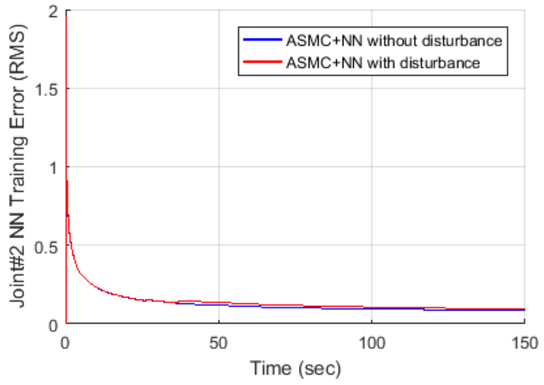


Fig. 7: Joint 1 and 2 NN training RMS.

5. Conclusion

In this paper, a new methodology to combine the neural network technique with adaptive SMC controller proposed. The simulation results and experimental results show that the proposed method can learn the dynamic of the robot and control the robot manipulator in the presence of model uncertainties and external disturbances and achieve high performance for the unforeseen trajectories. The results also show the proposed method has the capability to control manipulator under external disturbance. In future study, instead of using a three-layer neural network, the deep neural network will be employed to investigate the learning capability of the controller for more complex tasks such as object manipulation. The stability analysis is included in another work which will be presented in future.

Acknowledgements

The authors would like to acknowledge the financial support of Research, Development & Creativity Office (RDC Office) of Laurentian University.

References

[1] M. Zeinali, and L. Notash, “Adaptive sliding mode control with uncertainty estimator for robot manipulators, *Journal Of Mechanism And Machine Theory*, vol. 45, no. 1, pp. 80-90, 2010.

[2] S. Schaal, and C. G.Atkeson, “Learning Control in Robotics,” *IEEE Robotics & Automation Magazine*, vol. 17, pp. 20-29, 2010.

[3] M. Zeinali, “Adaptive Chattering-Free Sliding Mode Control Design Using Fuzzy Model of the System and Estimated Uncertainties and its Application to Robot Manipulators,” *IEEE*, Istanbul, Turkey, pp. 1-6, 2015. DOI: 10.1109/RASM.2015.7154652,

[4] O. Cerman, and P. Husek, “Adaptive fuzzy sliding mode control for electro-hydraulic servo mechanism,” *Expert Systems with Applications*, vol. 39, pp. 10269-10277, 2012.

[5] C. S. Chen, “Dynamic structure neural fuzzy networks for robust adaptive control of robot manipulators,” *IEEE Transactions of Industrial Electronics*, vol. 55 ,no. 9, pp. 3402-3414, 2008.

[6] D. Xu, D. Zhao, J. Yi and X. Tan, “Trajectory tracking control of omnidirectional wheeled mobile manipulators: Robust neural network based sliding mode approach,” *IEEE Trans. Systems Man and Cybernetics*, vol. 39, no. 3, pp. 788-799, 2009.

[7] J. Kang, W. Meng, A. Abraham and H. Liu, “An adaptive PID neural network for complex nonlinear system control,” Neurocomputing, vol. 135, 2014, pp. 79-85.

[8] Nabhan, T. M., and Zomaya, A. Y., Toward generating neural network structures for function approximation, Neural Networks, Vol. 7, No. 1, pp. 89-99, 1994.

[9] M. Zeinali, and L. Notash, “Robust adaptive neural fuzzy controller with model uncertainty estimator for manipulators,” *Transactions of the Canadian Society for Mechanical Engineering* (Trans. Can. Soc. Mech. Eng.), vol. 28, no. 2A, pp. 197-219, 2004.

[10] K. Hornik, M. Stinchcombe, H. Wite, “Multilayered feedforward network are universal approximator,” *Neural Network*, vol. 2, no. 5, pp. 359-366, 1989.
[11] J.-J. E. Slotine, and W. Li, *Applied Nonlinear Control*. Prentice-Hall Inc., 1991.
[12] D. H. Nguyen, and B. Widrow, “Neural Networks for self-learning control systems,” *IEEE Control Systems Mag.*, pp. 18-23, 1990.
[13] J. Angeles, *Fundamental of robotic mechanical systems, theory, methods, and algorithms*, 2nd ed. Springer, 2003.
[14] A. Fazeli, M. Zeinali, and A. Khajepour, “Application of Adaptive Sliding Mode Control for Regenerative Braking Torque Control of Air-Hybrid Engine,” *IEEE/ASME Transactions On Mechatronics*, no. 99, pp. 745-755, 2012. DOI: 10.1109/TMECH.2011.2129525.