Minimal Resource Allocating Networks for Discrete Time Sliding Mode Control of Robotic Manipulators

Maria Letizia Corradini, *Senior Member, IEEE*, Valentino Fossi, Andrea Giantomassi, Gianluca Ippoliti, *Member, IEEE*, Sauro Longhi, *Senior Member, IEEE*, and Giuseppe Orlando

***Abstract*—This paper presents a discrete-time sliding mode control based on neural networks designed for robotic manipulators. Radial basis function neural networks are used to learn about uncertainties affecting the system. The online learning algorithm combines the growing criterion and the pruning strategy of the minimal resource allocating network technique with an adaptive extended Kalman filter to update all the parameters of the networks. A method to improve the run-time performance for the real-time implementation of the learning algorithm has been considered. The analysis of the control stability is given and the controller is evaluated on the ERICC robot arm. Experiments show that the proposed controller produces good trajectory tracking performance and it is robust in the presence of model inaccuracies, disturbances and payload perturbations.**

***Index Terms*—Adaptive filters, discrete-time sliding mode control, minimal resource allocating networks, nonlinear systems, radial basis function networks, robotic manipulators, robust control.**

I. Introduction

ROBOTIC manipulators are highly nonlinear dynamic systems with unmodeled dynamics and uncertainties [1] that, being commonly used in industrial tasks, are expected to maintain good dynamic performance. The design of ideal controllers for such systems is a challenge for control engineers, mainly because of the nonlinearities and the coupling effects typical of robotic systems. Different approaches have been followed in order to cope with this problem, such as, for instance, feedback linearization [2], [3], model predictive control [4], [5], advanced PID control [6], adaptive [7] and sliding mode control [8]–[11]. In general, control approaches unable to guarantee some robustness can make the performance of the system, in terms of convergence, quite poor. As discussed in [12], global feedback linearization is possible in theory, but is difficult to achieve in practice as a consequence of uncertainties deriving from incomplete knowledge of the kinematics and dynamics, from joint and link flexibility, actuator dynamics, friction, sensor noise, and unknown loads. This imposes to couple the inverse dynamics approach with robust control methodologies [2], [13].

It is well known that sliding mode methods provide noticeable robustness and invariance properties to matched uncertainties [14]–[16], and they are computational simpler with respect to other robust control approaches. Recent literature contains a number of results about sliding mode control of manipulators, in some cases coupled with fuzzy control and or neuro-fuzzy techniques [17]–[23]. The largest part of these papers, however, uses the continuous-time dynamic model of the manipulator to design, leaving not addressed the issue of digitalization of the control law.

Digital control systems are currently receiving considerable credit as a consequence of the recent advances in digital microprocessor technology, and relevant interest is currently growing in the design of controllers based on the discrete model of the system. Nevertheless, the discrete-time counterpart of sliding mode control design has received only a limited attention [24]–[29]. Indeed, compared with continuous time sliding-mode strategies, the design problem in discrete-time has received much less coverage in the literature. This is due to its major drawback, consisting in the presence of a sector, of width depending on the available bound on uncertainties, where robustness is lost because the sliding mode condition cannot be exactly imposed. For this reason, only ultimate boundedness of trajectories can be guaranteed, and the larger are the uncertainties affecting the system, the wider is the bound on trajectories which can be guaranteed. As a possible solution to this problem, this paper proposes the design of the discontinuous control law, within the sector, based on an estimation, as accurate as possible, of uncertainties affecting the system.

Owing to their learning capabilities and universal approximation properties [30], Neural networks (NNs) will be used here to perform this approximation. It is well known in fact, that the learning ability of neural networks has been widely utilized in different industrial applications [31]–[38] and in particular to make controllers learn nonlinear characteristics of robots through experimental data, without a prior knowledge of their parameters and structure [39], [40]. Early NN-based control schemes for robotic manipulators produced good simulations or even experimental results [41], [42]. More recently, stable neural network control schemes have been investigated, such as nonlinearly parameterized NN-based adaptive control [43], [44] and linearly parameterized NN-based adaptive control [45], [46] for robotic manipulators. All these results proved that the

Manuscript received August 27, 2011; revised February 03, 2012; accepted April 02, 2012. Date of publication June 20, 2012; date of current version October 18, 2012. Paper No. TII-11-450.

M. L. Corradini is with the Scuola di Scienze e Tecnologie, Università di Camerino, 62032 Camerino (MC), Italy (e-mail: letizia.corradini@unicam.it).

V. Fossi, A. Giantomassi, G. Ippoliti, S. Longhi, and G. Orlando are with the Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, 60131 Ancona, Italy (e-mail: valentino.fossi@gmail.com; a.giantomassi@univpm.it; gianluca.ippoliti@univpm.it; sauro.longhi@univpm.it; giuseppe.orlando@univpm.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2012.2205395

stable NN-based control have the potential to deal with the difficulties for the control of robotic manipulators with unmodeled dynamics and uncertainties. The two books [47], [48] provide a good review of neural networks for the control of robotic manipulators.

In this paper, an online learning procedure is proposed to estimate the uncertainties affecting the system and radial basis function networks (RBFNs) have been considered for this estimation. These networks have been widely used for nonlinear system identification [49], [50] because of they have the ability both to approximate complex nonlinear mappings directly from input-output data with a simple topological structure that avoid lengthy calculations [50] and to reveal how learning proceeds in an explicit manner [51], [52]. The considered on-line learning algorithm is based on the extended minimal resource allocating network (EMRAN) technique [51]–[53], that adds hidden neurons to the network based on the innovation of each new RBFN input pattern which arrives sequentially. As stated in [51], [52], to obtain a more parsimonious network topology, a pruning strategy is introduced. This strategy detects and removes as learning progresses those hidden neurons which make little contribution to the network output. If an observation has no novelty then the existing parameters of the network are adjusted by an extended Kalman filter (EKF) [52], [54]. In this paper, the performance of the filter is improved by an on-line adjustment of the noise statistics obtained by a suitably defined estimation algorithm; the proposed adaptive extended Kalman filter (AEKF) is able to adaptively estimate the unknown statistical parameters [55], [56]. To minimize the computational effort for real-time implementation, a "winner neuron" strategy is incorporate in the learning algorithm [52].

The paper is organized as follows. The on-line learning algorithm is described in Section II. In Section III details on the considered control are discussed. Results on robot arm experimental tests are reported in Section IV. The paper ends with comments on the performance of the proposed controller.

II. Learning Algorithm

The standard approach to implement an extended minimal resource allocating network (EMRAN) is based on a sequential learning algorithm and an extended Kalman filter (EKF) [52], [54]. In particular the sequential learning algorithm adds and removes neurons online to the network according to a given criterion [51]–[53] and an EKF is used to update the net parameters. For the real-time implementation of the considered algorithm the computational load has been reduced by a "winner neuron" strategy [52]. In this paper the EMRAN algorithm is improved by an AEKF in order to take into account the time-varying noise statistics [55], as shown in the following of this section.

A. Radial Basis Function Neural Network

A RBFN with input pattern $\boldsymbol{\chi} \in \mathbb{R}^m$ and an output $\hat{\mathbf{y}} \in \mathbb{R}^n$ implements a mapping $\mathbf{h} : \mathbb{R}^m \rightarrow \mathbb{R}^n$ according to

$$\hat{\boldsymbol{y}} = \boldsymbol{h}(\boldsymbol{\chi}) = \boldsymbol{\lambda}_0 + \sum_{z=1}^{L} \boldsymbol{\lambda}_z \phi(\|\boldsymbol{\chi} - \boldsymbol{c}_z\|) \tag{1}$$

where $\phi(\cdot)$ is a given function from $\mathbb{R}^+$ to $\mathbb{R}^n$, $\|\cdot\|$ denotes the Euclidean norm, $\boldsymbol{\lambda}_z \in \mathbb{R}^n$, $z = 0, 1, \ldots, L$ are the weights or parameters, $\boldsymbol{c}_z \in \mathbb{R}^m$, $z = 1, 2, \ldots, L$, are the radial basis functions centers (called also units or neurons) and L is the number of centers [50]. The terms

$$\boldsymbol{o}_z = \boldsymbol{\lambda}_z \phi(\|\boldsymbol{\chi} - \boldsymbol{c}_z\|),\ z = 1, \ldots, L \tag{2}$$

with $\boldsymbol{o}_z \in \mathbb{R}^n$, are called the hidden unit outputs.

The RBFN is used for the estimation of the uncertainties affecting the robotic system. The uncertainty dynamics can be taken into account through the network input pattern $\boldsymbol{\chi}$, that must be composed of a proper set of system input and output samples acquired in a finite set of past time instants [57] as specified in (33).

Theoretical investigation and practical results show that the choice of the nonlinearity $\phi(\cdot)$, a function of the distance d_z between the current input $\boldsymbol{\chi}$ and the centre $\boldsymbol{c}_z$, does not significantly influence the performance of the RBFN [50]. Therefore, the following gaussian function is considered:

$$\phi(d_z) = \exp\left(\frac{-d_z^2}{\beta_z^2}\right), \quad z = 1, 2, \ldots, L \tag{3}$$

where $d_z = \|\boldsymbol{\chi} - \boldsymbol{c}_z\|$ and the real constant β_z is a scaling or "width" parameter [50].

For the further analysis of Section III-B, the following assumption reflecting the universal approximation capability of NNs is made [30], [58]:

Assumption 2.1: The RBFN input pattern $\boldsymbol{\chi} \in \mathbb{R}^m$ is contained within a known, arbitrary large compact set $\boldsymbol{\Omega} \subset \mathbb{R}^m$. Inside this compact set, exist an optimal set of weights $\boldsymbol{\lambda}_z^* \in \mathbb{R}^n$, $z = 0, 1, \ldots, L$ such that the approximation error of the optimal neural network, with enough centers and the same structure of (1), is arbitrarily small.

B. Minimal Resource Allocating Network Algorithm

The learning process of EMRAN involves allocation of new hidden units, pruning and "winner neuron" strategies as well as adaptation of network parameters [52]–[54]. The network starts with no hidden units and as input-output data $(\boldsymbol{\chi}(\cdot), \boldsymbol{y}(\cdot))$ are received, where $\boldsymbol{y}(\cdot)$ is the desired output of the net, some of them are used to generate new hidden units based on a suitably defined growth criteria. In particular at each time instant k the following three conditions are evaluated to decide if the input $\boldsymbol{\chi}(k)$ should give rise to a new hidden unit:

$$\|\boldsymbol{e}(k)\| = \|\boldsymbol{y}(k) - \boldsymbol{h}(\boldsymbol{\chi}(k)\| > E_1 \tag{4}$$

$$e_{\text{rms}}(k) = \sqrt{\sum_{j=k-(M-1)}^{k} \frac{\|\boldsymbol{e}_j(k)\|^2}{M}} > E_2 \tag{5}$$

$$d(k) = \|\boldsymbol{\chi}(k) - \boldsymbol{c}_r(k)\| > E_3 \tag{6}$$

where $\boldsymbol{e}(k) = [e_1(k), \ldots, e_n(k)]^T$ is the residual vector, $\boldsymbol{c}_r(k)$ is the centre of the hidden unit that is nearest to $\boldsymbol{\chi}(k)$ and M represents the number of past network outputs to calculate the output error $e_{\text{rms}}(k)$. The terms E_1, E_2, and E_3 are thresholds

to be suitably selected. As stated in [51], [52], these three conditions evaluate the novelty in the data. If all the criteria of (4)–(6) are satisfied, a new hidden unit is added and the following parameters are associated with it:

$$\boldsymbol{\lambda}_{L+1} = \boldsymbol{e}(k) \tag{7}$$

$$\boldsymbol{c}_{L+1} = \boldsymbol{\chi}(k) \tag{8}$$

$$\beta_{L+1} = \gamma \|\boldsymbol{\chi}(k) - \boldsymbol{c}_r(k)\| \tag{9}$$

where γ determines the overlap of the response of a hidden unit in the input space as specified in [52] and [54]. If the observation $(\boldsymbol{\chi}(k), \boldsymbol{y}(k))$ does not satisfy the criteria of (4)–(6), an EKF is used to update the following parameters of the network:

$$\boldsymbol{w} = \left[\boldsymbol{\lambda}_0^T, \boldsymbol{\lambda}_1^T, \boldsymbol{c}_1^T, \beta_1, \ldots, \boldsymbol{\lambda}_L^T, \boldsymbol{c}_L^T, \beta_L\right]^T. \tag{10}$$

The updating equation is given by

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) + \boldsymbol{K}(k)\boldsymbol{e}(k) \tag{11}$$

where the gain matrix $\boldsymbol{K}(k)$ is expressed by

$$\boldsymbol{K}(k) = \boldsymbol{P}(k-1)\boldsymbol{H}(k)[\boldsymbol{R}(k) + \boldsymbol{H}^T(k)\boldsymbol{P}(k-1)\boldsymbol{H}(k)]^{-1} \tag{12}$$

with $\boldsymbol{H}(k)$ the gradient matrix of the function $\boldsymbol{h}(\boldsymbol{\chi}(k))$ with respect to the parameter vector $\boldsymbol{w}(k-1)$ [52] and [54], $\boldsymbol{R}(k)$ is the variance of the measurement noise and $\boldsymbol{P}(k-1)$ is the error covariance matrix. This matrix is updated by

$$\boldsymbol{P}(k) = \left[\boldsymbol{I}_{\nu\times\nu} - \boldsymbol{K}(k)\boldsymbol{H}^T(k)\right]\boldsymbol{P}(k-1) + \sigma_\eta^2(k-1)\boldsymbol{I}_{\nu\times\nu} \tag{13}$$

where $\boldsymbol{I}$ is the identity matrix and $\sigma_\eta^2(k-1)$ is introduced to avoid that the rapid convergence of the EKF algorithm prevents the model from adapting to future data [52], [54]. The $\nu \times \nu$ matrix $\boldsymbol{P}(k)$ is positive definite symmetric and ν is the number of parameters to be adjusted. When a new hidden neuron is allocated, the dimension of $\boldsymbol{P}(k)$ increases as follows to take into account the new parameters introduced by adding the new hidden neuron [52]:

$$\boldsymbol{P}(k) = \begin{pmatrix} \boldsymbol{P}(k-1) & 0 \\ 0 & p_0\boldsymbol{I}_{\nu_1\times\nu_1} \end{pmatrix}. \tag{14}$$

In (14), p_0 is an estimate of the uncertainty in the initial values assigned to the parameters and the dimension ν_1 of the identity matrix $\boldsymbol{I}$ is the number of new parameters introduced by adding the new hidden neuron. As stated in [51] and [52], to keep the RBF network in a minimal size a pruning strategy removes those hidden units that contribute little to the overall network outputs over a number of consecutive observations. To carry out this pruning strategy, for each observation $(\boldsymbol{\chi}(k), \boldsymbol{y}(k))$ the vectors $\boldsymbol{o}_z \in \mathbb{R}^n$, $z = 1, \ldots, L$ of hidden unit outputs are computed [see (2)] and each element o_{zi}, $z = 1, \ldots, L$; $i = 1, \ldots, n$ of these vectors is normalized with respect to the highest element of all vectors

$$\bar{o}_{zi}(k) = \frac{o_{zi}(k)}{\max\{o_{1i}(k), \ldots, o_{Li}(k)\}} \tag{15}$$

with $z = 1, \ldots, L$ and $i = 1, \ldots, n$. The hidden units for which all their normalized elements (15) are less than a threshold δ for ξ consecutive observations are removed and the dimensionality of all the related matrices are adjusted to suit the reduced network [51], [52].

The weakness of the above algorithm is that all the parameters of the network, including all the centers of the hidden neurons, widths and weights, have to be update at every step; the size of the matrices to be update becomes large as the number of hidden neurons increases. Therefore, for the real-time implementation of the considered algorithm, it is necessary to reduce the online computation effort and to this purpose a "winner neuron" strategy is incorporate in the learning algorithm [52]. The "winner neuron" is defined as the neuron in the network that is closest (in some norm sense) to the current input data. The criteria for adding and pruning the hidden neurons are all the same as in the above algorithm; the difference, in the "winner neuron" strategy, is that if the observation $(\boldsymbol{\chi}(\cdot), \boldsymbol{y}(\cdot))$ does not meet the criteria to add a new hidden neuron [see (4)–(6)], only the network parameters $\boldsymbol{w}^* \subseteq \boldsymbol{w}$ related to the selected "winner neuron" are updated by the EKF algorithm. In particular, the updating (11) becomes

$$\boldsymbol{w}^*(k) = \boldsymbol{w}^*(k-1) + \boldsymbol{K}^*(k)\boldsymbol{e}(k) \tag{16}$$

with

$$\boldsymbol{w}^* = \left[\boldsymbol{\lambda}_0^T, \boldsymbol{\lambda}^{*T}, \boldsymbol{c}^{*T}, \beta^*\right]^T. \tag{17}$$

In (16), the gain matrix $\boldsymbol{K}^* \subseteq \boldsymbol{K}$ is expressed by

$$\boldsymbol{K}^*(k) = \boldsymbol{P}^*(k-1)\boldsymbol{H}^*(k)\Big[\boldsymbol{R}(k) + \boldsymbol{H}^{*T}(k)\boldsymbol{P}^*(k-1)\boldsymbol{H}^*(k)\Big]^{-1} \tag{18}$$

with $\boldsymbol{H}^*(k)$ the gradient matrix of the function $\boldsymbol{h}(\boldsymbol{\chi}(k))$ with respect to the parameter vector $\boldsymbol{w}^*(k-1)$ and $\boldsymbol{P}^* \subseteq \boldsymbol{P}$ is the error covariance matrix which is updated by

$$\boldsymbol{P}^*(k) = \left[\boldsymbol{I}_{\nu^*\times\nu^*} - \boldsymbol{K}^*(k)\boldsymbol{H}^{*T}(k)\right]\boldsymbol{P}^*(k-1) + \sigma_\eta^2(k-1)\boldsymbol{I}_{\nu^*\times\nu^*} \tag{19}$$

where ν^* is the number of parameters to be adjusted.

The EKF can be implemented once estimates of $\sigma_\eta^2(k)$ and $\boldsymbol{R}(k)$ are available. In general, a complete and reliable information about these estimates is not available; on the other hand it is well known how poor knowledge of noise statistics may seriously degrade the Kalman filter performance. This problem is here dealt with introducing an adaptive adjustment mechanism of $\sigma_\eta^2(k)$ and $\boldsymbol{R}(k)$ values in the EKF equations.

C. *Adaptive Estimation of* $\sigma_\eta^2(k)$ *and* $\boldsymbol{R}(k)$

A considerable amount of research has been performed on the adaptive Kalman filtering [55], [59]–[62], but in practice it is often necessary to redesign the adaptive filtering scheme according to the particular characteristics of the problem faced. The adaptive procedure here proposed refers to $\boldsymbol{R}(k) = \operatorname{diag}\left[\sigma_{v,1}^2(k), \ldots, \sigma_{v,n}^2(k)\right]$. Under proper assumptions given in [55], it is possible to define a simple and efficient estimation algorithm based on the condition of consistency, at each step, between the residuals $e_i(k)$, $i = 1, \ldots, n$, and

their predicted statistic $E\left\{e_i^2(k)\right\}$, $i = 1, \ldots, n$. Imposing such a condition, one-stage estimates $\hat{\sigma}_\eta^2(k-1)$ and $\hat{\sigma}_{v,i}^2(k)$, $i = 1, \ldots, n$, of $\sigma_\eta^2(k-1)$ and $\sigma_{v,i}^2(k)$, $i = 1, \ldots, n$, respectively, are obtained at each step. To increase their significance, the one-stage estimates $\hat{\sigma}_\eta^2(k-1)$ and $\hat{\sigma}_{v,i}^2(k)$, $i = 1, \ldots, n$, are average obtaining the relative smoothed version $\bar{\hat{\sigma}}_\eta^2(k-1)$ and $\bar{\hat{\sigma}}_{v,i}^2(k)$, $i = 1, \ldots, n$. After proper calculations [55], the following recursive form of estimates $\bar{\hat{\sigma}}_\eta^2(k-1)$ and $\bar{\hat{\sigma}}_{v,i}^2(k)$, $i = 1, \ldots, n$, is found:

$$\bar{\hat{\sigma}}_\eta^2(k-1) = \bar{\hat{\sigma}}_\eta^2(k-2) + \frac{1}{(l_\eta+1)n}\left[\sum_{i=1}^{n}\left(\hat{\sigma}_{\eta,i}^2(k-2) - \hat{\sigma}_{\eta,i}^2(k-(l_\eta+1))\right)\right] \tag{20}$$

$$\bar{\hat{\sigma}}_{v,i}^2(k) = \bar{\hat{\sigma}}_{v,i}^2(k-1) + \frac{1}{l_v+1}(\hat{\sigma}_{v,i}^2(k) - \hat{\sigma}_{v,i}^2(k-l_v)), i = 1, \ldots, n \tag{21}$$

where

- $\hat{\sigma}_{\eta,i}^2(k-1) = \max\left\{\left(\boldsymbol{H}_i^T(k)\boldsymbol{H}_i(k)\right)^{-1}\left[e_i(k)^2 - \boldsymbol{H}_i^T(k)\boldsymbol{P}(k-1)\boldsymbol{H}_i(k) - \bar{\hat{\boldsymbol{\sigma}}}_{v,i}^2(k)\right], 0\right\}$, $i = 1, \ldots, n$
- $\hat{\sigma}_{v,i}^2(k) = \max\left\{e_i^2(k) - [\boldsymbol{H}_i^T(k)\boldsymbol{P}(k-1)\cdot \boldsymbol{H}_i(k) + \boldsymbol{H}_i^T(k)\bar{\hat{\sigma}}_\eta^2(k-1)\boldsymbol{I}\boldsymbol{H}_i(k)], 0\right\}$, $i = 1, \ldots, n$
- l_η and l_v are the number of one-stage estimates $\bar{\hat{\sigma}}_\eta^2(k-1)$ and $\bar{\hat{\sigma}}_{v,i}^2(k)$, $i = 1, \ldots, n$, respectively, yielding the smoothed estimates.

Parameters l_η and l_v of estimators (20) and (21) are chosen on the basis of two antagonist considerations: low values would produce noise estimators which are not statistically significant, large values would produce estimators which are scarcely sensitive to possible rapid fluctuations of the true $\sigma_\eta^2(k-1)$ and $\sigma_{v,i}^2(k)$, $i = 1, \ldots, n$ [55]. In other words, the one-stage estimates are made by averaging past samples in order to increase the statistical significance of estimators; if the samples are too far the filter has a low reactivity, while if the samples are too near estimators have a low statistical significance [55]. During filter initialization, the starting values $\hat{\sigma}_\eta^2(0)$ and $\hat{\sigma}_{v,i}^2(0)$, $i = 1, \ldots, n$, in (20) and (21), respectively, must be chosen on the basis of the *a priori* available information. In case of lack of such information, a large value of $P(0,0)$ is useful to prevent divergence.

The EMRAN estimation algorithm [51], [52] enhanced by the AEKF, called EMRANAEKF algorithm, is summarized as follows:

1. For each observation $(\boldsymbol{\chi}(k), \boldsymbol{y}(k))$ do: compute the overall network output: $\hat{\boldsymbol{y}}(k) = \boldsymbol{h}(\boldsymbol{\chi}(k)) = \boldsymbol{\lambda}_0 + \sum_{z=1}^{L}\boldsymbol{\lambda}_z\phi\left(\|\boldsymbol{\chi}(k) - \boldsymbol{c}_z(k)\|\right)$ where L is the number of hidden units;
2. Calculate the parameters required by the growth criterion
   — $\|\boldsymbol{e}(k)\| = \|\boldsymbol{y}(k) - \boldsymbol{h}(\boldsymbol{\chi}(k))\|$
   — $e_{\text{rms}}(k) = \sqrt{\sum_{j=k-(M-1)}^{k}(\|\boldsymbol{e}_j(k)\|^2/M)}$
   — $d(k) = \|\boldsymbol{\chi}(k) - \boldsymbol{c}_r(k)\|$
3. Apply the criterion for adding a new hidden unit: **if** $\|\boldsymbol{e}(k)\| > E_1$ **and** $e_{\text{rms}}(k) > E_2$ **and** $d(k) > E_3$ **then** allocate a new hidden unit $L+1$ with
   — $\boldsymbol{\lambda}_{L+1} = \boldsymbol{e}(k)$
   — $\boldsymbol{c}_{L+1} = \boldsymbol{\chi}(k)$
   — $\beta_{L+1} = \gamma\|\boldsymbol{\chi}(k) - \boldsymbol{c}_r(k)\|$
   else
   — adapt process or measurement noise coefficients as stated in [55]: $\bar{\hat{\sigma}}_\eta^2(k-1) = \bar{\hat{\sigma}}_\eta^2(k-2) + (1/(l_\eta+1)n)\left[\sum_{i=1}^{n}\left(\hat{\sigma}_{\eta,i}^2(k-2) - \hat{\sigma}_{\eta,i}^2(k-(l_\eta+1))\right)\right]$, $i = 1, \ldots, n$
   $\bar{\hat{\sigma}}_{v,i}^2(k) = \bar{\hat{\sigma}}_{v,i}^2(k-1) + (1/l_v+1)(\hat{\sigma}_{v,i}^2(k) - \hat{\sigma}_{v,i}^2(k-l_v))$, $i = 1, \ldots, n$
   — update only network parameters $\boldsymbol{w}^*$ related to the selected "winner neuron": $\boldsymbol{w}^*(k) = \boldsymbol{w}^*(k-1) + \boldsymbol{K}^*(k)\boldsymbol{e}(k)$
   — update the error covariance matrix related to the selected "winner neuron": $\boldsymbol{P}^*(k) = \left[\boldsymbol{I}_{\nu^*\times\nu^*} - \boldsymbol{K}^*(k)\boldsymbol{H}^{*T}(k)\right]\boldsymbol{P}^*(k-1) + \sigma_\eta^2(k-1)\boldsymbol{I}_{\nu^*\times\nu^*}$
   end
4. Check the criterion to prune hidden units:
   — compute the hidden unit outputs: $\boldsymbol{o}_z(k) = \boldsymbol{\lambda}_z\phi\left(\|\boldsymbol{\chi}(k) - \boldsymbol{c}_z(k)\|\right)$, $z = 1, \ldots, L$
   — compute the normalized outputs: $\bar{o}_{zi}(k) = o_{zi}(k)/\}$ $\max\{o_{1i}(k), \ldots, o_{Li}(k), z = 1, \ldots, L; i = 1, \ldots, n$
   — **if** $\bar{o}_{zi}(\cdot) < \delta$, $i = 1, \ldots, n$ for ξ consecutive observations **then** prune the zth hidden unit and reduce the dimensionality of the related matrices **end**
5. $k = k+1$ and **go** to step 1.

III. Control Design

A. Preliminaries

From the Euler-Lagrangian formulation, the equations of motion of a robot manipulator can be written as [63]

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{22}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector of generalized coordinates (rotational joint configurations), $\boldsymbol{B}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^n$ represents centrifugal and Coriolis torques, $\boldsymbol{F}_v \in \mathbb{R}^{n\times n}$ is the diagonal matrix of the viscous friction coefficients, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^n$ is the vector of gravitational torques and $\boldsymbol{\tau} \in \mathbb{R}^n$ is the vector of torques acting at the joints. As well known, the robot model (22) is characterized by the structural properties given in [63]. Introducing the state vector $\boldsymbol{x} = [x_1 \cdots x_n\ x_{n+1} \cdots x_{2n}]^T = \left[\boldsymbol{q}^T\ \dot{\boldsymbol{q}}^T\right]^T$, the control input $\boldsymbol{u} = \boldsymbol{\tau}$, and considering possible uncertainties affecting model (22), this latter can be expressed as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_c(\boldsymbol{x}) + \boldsymbol{g}_c(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{f}_c^0(\boldsymbol{x}) + \boldsymbol{\Delta f}_c(\boldsymbol{x}) + \left(\boldsymbol{g}_c^0(\boldsymbol{x}) + \boldsymbol{\Delta g}_c(\boldsymbol{x})\right)\boldsymbol{u} \tag{23}$$

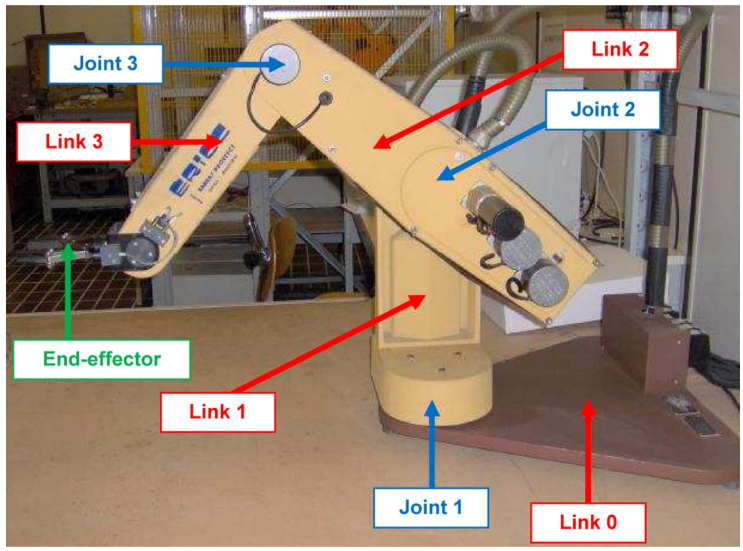


Fig. 1. ERICC manipulator.

where $\Delta \boldsymbol{f}_c(\boldsymbol{x})$, $\Delta \boldsymbol{g}_c(\boldsymbol{x})$ depend on the uncertainties, while the nominal model is given by $\boldsymbol{f}_c^0(\boldsymbol{x})$, $\boldsymbol{g}_c^0(\boldsymbol{x})$

$$\boldsymbol{f}_c^0(\boldsymbol{x}) = \begin{bmatrix} x_{n+1} \\ \vdots \\ x_{2n} \\ f_1(\boldsymbol{x}) \\ \vdots \\ f_n(\boldsymbol{x}) \end{bmatrix}; \; \boldsymbol{g}_c^0(\boldsymbol{x}) = \begin{bmatrix} & \mathbf{o}_{n\times n} & \\ g_{1,1}(\boldsymbol{x}) & \cdots & g_{1,n}(\boldsymbol{x}) \\ \cdots & \cdots & \cdots \\ g_{n,1}(\boldsymbol{x}) & \cdots & g_{n,n}(\boldsymbol{x}) \end{bmatrix}. \tag{24}$$

Note that all the terms present in (23) and (24) can be easily computed from (22).

Assumption 3.1: In view of the existence of physical bounds on achievable positions and velocities by the robot arm, it is assumed that the uncertain terms $\Delta \boldsymbol{f}_c(\boldsymbol{x})$, $\Delta \boldsymbol{g}_c(\boldsymbol{x})$ are norm bounded.

A planar two-link manipulator with revolution joints [63] will be considered in this paper, in order to illustrate the feasibility of the proposed control algorithm. Therefore, the variable $\boldsymbol{q}$ is $\boldsymbol{q} = [q_2 \; q_3]^T$, where q_2, q_3 denote the joint displacements of the two considered rotational joints 2 and 3 of Fig. 1. The arm dynamics is described by (22) with $n = 2$ and the detailed model can be found in [63] and [64].

B. Sliding Mode Controller Design

In this section, the development of a NN-based discrete-time sliding mode control (NNSMC) law is described, aimed at solving the trajectory tracking problem in the joint space of the considered planar two-link manipulator.

Control design will be carried out in the discrete time framework, and discretization after control design (performed in the continuous-time framework) will be avoided, in accordance to the discussion reported in [65]. In this sense, the design approach used belongs to the so called "classical" sliding mode design techniques in the framework of discrete-time sliding modes [26], [66]. In this context, several approaches are available in literature for the discretization of a linear plant using ZOH method [67]–[69], showing that inherent properties of SMC are not maintained after discretization. However, due to the presence of strong nonlinearities in the robot model, a simpler approach has to be preferred for the plant discretization, using the Euler method. Finally, consider that the plant discretization method is likely not to seriously affect closed loop performances, since the control design is performed directly in the discrete time domain [65].

Considering a sampling time T_c, and discretizing the uncertain model (23) by Euler method, one has

$$\begin{cases} \boldsymbol{q}_s(k+1) = \boldsymbol{q}_s(k) + T_c \dot{\boldsymbol{q}}_s(k) \\ \dot{\boldsymbol{q}}_s(k+1) = \dot{\boldsymbol{q}}_s(k) + T_c \boldsymbol{f}(k) + T_c \boldsymbol{g}(k)\boldsymbol{u}(k) + \boldsymbol{n}(k) \end{cases} \tag{25}$$

with

$$\begin{aligned} \boldsymbol{q}_s(k) &= [x_1(kT_c) \; x_2(kT_c)]^T \\ \dot{\boldsymbol{q}}_s(k) &= [x_3(kT_c) \; x_4(kT_c)]^T. \end{aligned}$$

Moreover

$$\boldsymbol{f}(k) = [f_1(k) \; f_2(k)]^T, \quad \boldsymbol{g}(k) = \begin{bmatrix} g_{1,1}(k) \; g_{1,2}(k) \\ g_{2,1}(k) \; g_{2,2}(k) \end{bmatrix}$$

[see (24) for $n = 2$], where with some abuse of notation we have written $f_i(k) = f_i(\boldsymbol{x}(kT_c))$, $i = 1, 2$, $g_{i,j}(k) = g_{i,j}(\boldsymbol{x}(kT_c))$, $i, j = 1, 2$. Finally, $\boldsymbol{n}(k)$ is given by

$$\boldsymbol{n}(k) = T_c(\Delta \boldsymbol{f}(k) + \Delta \boldsymbol{g}(k)\boldsymbol{u}(k)) \tag{26}$$

with

$$\Delta \boldsymbol{f}(k) = \Delta \boldsymbol{f}_c(\boldsymbol{x}(kT_c)), \quad \Delta \boldsymbol{g}(k) = \Delta \boldsymbol{g}_c(\boldsymbol{x}(kT_c)).$$

Assumption 3.2: It is assumed that $\boldsymbol{g}(k)$ and $\boldsymbol{g}(k) + \Delta \boldsymbol{g}(k)$ are invertible matrices $\forall \, \boldsymbol{x}(kT_c)$, $\forall \, k$ for the chosen T_c.

Remark 3.1: According to Assumption 3.1, the matrix $\boldsymbol{g}(k)$ and the uncertain terms $\boldsymbol{\Delta f}(k)$ and $\boldsymbol{\Delta g}(k)$ are bounded by known constants

$$\|\boldsymbol{g}(k)\| \le g_M; \quad \|\boldsymbol{\Delta f}(k)\| \le \rho_f;$$
$$\|\boldsymbol{\Delta g}(k)\| \le \rho_g; \quad \|\boldsymbol{g}(k) + \boldsymbol{\Delta g}(k)\| \ge g_{\min}.$$

The control law ensuring the robust tracking of a reference variable $\boldsymbol{q}_d(k) = [x_{1,d}(kT_c)\, x_{2,d}(kT_c)]^T$ by the sampled position $\boldsymbol{q}_s(k) = [x_1(kT_c)\, x_2(kT_c)]^T$ will be described in the following. Define the discrete-time tracking error as $\boldsymbol{\pi}(k) = [\pi_1(kT_c)\, \pi_2(kT_c)]^T = \boldsymbol{q}_s(k) - \boldsymbol{q}_d(k)$, and consider the following discrete-time variable:

$$\boldsymbol{s}(k) = \boldsymbol{\pi}(k+1) - \boldsymbol{\Lambda}\boldsymbol{\pi}(k) \tag{27}$$

with $eig(\boldsymbol{\Lambda}) = \lambda_i$, $i = 1, 2$ such that $|\lambda_i| < 1$. Using (25), it can be shown that

$$\boldsymbol{s}(k) = (\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{q}_s(k) + T_c\dot{\boldsymbol{q}}_s(k) - \boldsymbol{q}_d(k+1) + \boldsymbol{\Lambda}\boldsymbol{q}_d(k) \tag{28}$$

Remark 3.2: Note that $\boldsymbol{s}(k)$ is always computable at the time instant k. In fact, from (25) the term $\boldsymbol{q}_s(k+1)$ present in (27) can be replaced by $\boldsymbol{q}_s(k+1) = \boldsymbol{q}_s(k) + T_c\dot{\boldsymbol{q}}_s(k)$, producing (28).

Moreover, consider the following sliding surface:

$$\boldsymbol{\sigma}(k) = \boldsymbol{s}(k) - \alpha\boldsymbol{s}(k-1) = \boldsymbol{0}; \quad 0 < |\alpha| < 1. \tag{29}$$

If an ideal sliding mode is achieved on (29), variable $\boldsymbol{s}(k)$ asymptotically converges to zero. As a consequence, due to (27), the tracking error $\boldsymbol{\pi}(k)$ is asymptotically vanishing. The following result shows that the achievement of a quasi sliding motion on the surface (29) implies that the tracking error can be made arbitrarily small in norm.

Theorem 3.1: Consider the arm model (25) and Remark 3.1. A quasisliding motion on the surface (29) is enforced by the control law $\boldsymbol{u}(k) = \boldsymbol{u}^{eq}(k) + \boldsymbol{u}^n(k)$, with

$$\begin{aligned} T_c^2\boldsymbol{g}(k)\boldsymbol{u}^{eq}(k) = &-(\boldsymbol{I} - \boldsymbol{\Lambda})\boldsymbol{q}_s(k) - T_c(2\boldsymbol{I} - \boldsymbol{\Lambda})\dot{\boldsymbol{q}}_s(k) \\ &-T_c^2\boldsymbol{f}(k) - \boldsymbol{\Lambda}\boldsymbol{q}_d(k+1) + \boldsymbol{q}_d(k+2) + \alpha\boldsymbol{s}(k) \end{aligned} \tag{30}$$

and

$$\boldsymbol{u}^n(k) = \begin{cases} \theta(\|\tilde{\boldsymbol{\sigma}}(k)\| - \rho_s) \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, & \text{if } \|\tilde{\boldsymbol{\sigma}}(k)\| - \rho_s > 0 \\ -\left[\boldsymbol{g}(k)T_c^2\right]^{-1}\hat{\boldsymbol{n}}(k), & \text{otherwise} \end{cases} \tag{31}$$

with $0 < |\theta| < 1$, $0 < |\alpha| < 1$ and

$$\tilde{\boldsymbol{\sigma}}(k) = \frac{\boldsymbol{\sigma}(k)}{T_C^2(g_M + \rho_g)}; \quad \rho_s = \frac{\rho_f + \rho_g U_M}{g_{\min}} \tag{32}$$

where U_M is the maximum input torque supplied by joint actuators, i.e., $\|\boldsymbol{u}\| \le U_M$. The approximation $\hat{\boldsymbol{n}}(k)$ of $\boldsymbol{n}(k)$, is performed by a neural network $\boldsymbol{h} : \mathbb{R}^6 \to \mathbb{R}^2$ of the form (1) with the output $\hat{\boldsymbol{y}}(k) := \hat{\boldsymbol{n}}(k) \in \mathbb{R}^2$ and the input pattern $\boldsymbol{\chi} \in \mathbb{R}^6$ defined as

$$\boldsymbol{\chi}(k) := [\boldsymbol{q}_s(k)\, \boldsymbol{u}^n(k-1)\, \boldsymbol{n}(k-1)]\,. \tag{33}$$

The desired output of the network $\boldsymbol{y}(k) := \boldsymbol{n}(k) \in \mathbb{R}^2$ has the form given by (26). The algorithm used for net training minimizes the NN functional approximation errors $\boldsymbol{e}(k) := \boldsymbol{n}(k) - \hat{\boldsymbol{n}}(k)$ by the procedure described in Section II and under the Assumption 2.1.

Proof: Inserting (30) in $\boldsymbol{\sigma}(k+1)$ gives

$$\boldsymbol{\sigma}(k+1) = T_C^2\boldsymbol{g}(k)\boldsymbol{u}^n(k) + T_C\boldsymbol{n}(k). \tag{34}$$

The imposition of the condition $\|\boldsymbol{\sigma}(k+1)\| < \|\boldsymbol{\sigma}(k)\|$ produces, considering (26) and (34)

$$\|[\boldsymbol{g}(k) + \Delta\boldsymbol{g}(k)] \cdot [\boldsymbol{u}^n(k) + \boldsymbol{d}(k)]\| < \frac{\|\boldsymbol{\sigma}(k)\|}{T_c^2} \tag{35}$$

with $\boldsymbol{d}(k)$ given by

$$\boldsymbol{d}(k) = [\boldsymbol{g}(k) + \Delta\boldsymbol{g}(k)]^{-1}\,[\Delta\boldsymbol{f}(k) + \boldsymbol{\Delta g}(k)\boldsymbol{u}^{eq}(k)]\,. \tag{36}$$

Condition (35) is fulfilled if

$$\|\boldsymbol{u}^n(k) + \boldsymbol{d}(k)\| < \|\tilde{\boldsymbol{\sigma}}(k)\|. \tag{37}$$

Unlikely continuous-time sliding modes, for discrete-time systems the plant cannot be permanently restricted to the designed surface. What can be ensured is the following decreasing condition $\|\boldsymbol{\sigma}(k+1)\| < \|\boldsymbol{\sigma}(k)\|$, which unfortunately cannot be ensured $\forall\ k$, but can be guaranteed outside a given region. In fact, it is easy to verify that condition (37) is guaranteed by $\boldsymbol{u}^n(k)$ given in (31) when $\|\tilde{\boldsymbol{\sigma}}(k)\| > \rho_s$. On the contrary, when $\|\tilde{\boldsymbol{\sigma}}(k)\| \le \rho_s$, i.e., inside the sector, the sliding mode condition cannot be imposed exactly and an estimation $\hat{\boldsymbol{n}}(k)$ of $\boldsymbol{n}(k)$ is used, given by a neural network of the form (1). Replacing $\boldsymbol{n}(k)$ by $\hat{\boldsymbol{n}}(k)$ in (34), and setting $\boldsymbol{\sigma}(k+1) = \boldsymbol{0}$, control law (31) is obtained, for $\|\tilde{\boldsymbol{\sigma}}(k)\| \le \rho_s$. From a theoretical viewpoint, for the universal approximation capability of NNs (see Assumption 2.1), the error introduced by this approximation can be made arbitrarily small. The previous developments can be summarized as follows: the variable $\boldsymbol{\sigma}(k)$ tends to the region $\|\tilde{\boldsymbol{\sigma}}(k)\| \le \rho_s$ because of the choice of $\boldsymbol{u}^n(k)$ given in (31). Once such region is entered, it approximately holds $\boldsymbol{\sigma}(k+1) \simeq \boldsymbol{0}$ in view of the approximation capability of NNs. Because of the expression of the sliding surface (29), this implies that the tracking error is arbitrarily small in norm. ■

Remark 3.3: It is worth noticing that Assumption 2.1 is based on the possibility, only theoretical, to have a sufficiently large number of centers. Considering implementation aspects, it is reasonable to expect that the error norm is not arbitrarily but negligibly small.

IV. Experimental Implementation

The proposed controller has been implemented on an ERICC robot arm (see Fig. 1), built by Barras Provence, France. The robot is installed in the Robotics Laboratory at the Dipartimento di Ingegneria dell'Informazione, of the Università Politecnica

delle Marche. In Fig. 1, is shown the robot with labels indicating the three base joints. Other two joints (not indicated in Fig. 1) are for wrist movements. In this section, the experimental setup and results are discussed.

A. Experimental Setup

The considered robot has five degrees of freedom but for the sake of simplicity only links 2 and 3 have been utilized in the experiments. Anyway, the developed experimental validation over a real planar robot can give the feasibility of this industrial application. The two considered rotational joints 2 and 3 are actuated by two dc motors with reduction gears. Position measurements are obtained by means of potentiometers and velocity measurements by tachometers. The ERICC command module consists of a power supply module, which provides the servo power for the system; a joint interface module, which contains the hardware to drive the motors and provides sensor feedback from each joint; and a processor module to run user developed software. In order to implement complex control algorithms, a new controller is used in this setup in place of the original ERICC processor module. This system, including hardware and software, combines an experimental apparatus with an easy-to-use software platform based on a dSPACE controller board [70]. In particular the control law is implemented on a dSPACE DS1102 real-time controller board. A sampling time of 0.01 s has been used.

B. Structure and Validation of Implemented NNs

Training and testing phases of considered NNs have been performed online with data acquired on a set of planned trajectories chosen with different shapes and bounds on trajectory derivatives [71] and considering different payload configurations for the robot. In particular, a RBFN is designed to estimate uncertainties $\boldsymbol{n}(k) = [n_2(k)\ n_3(k)]^T$ of controlled links 2 and 3. The set of experimental data used to train the NNs is given by the pairs $(\boldsymbol{\chi}(k), \boldsymbol{n}(k))$, $k = 1, 2, \ldots$, where $\boldsymbol{\chi}(k)$ and $\boldsymbol{n}(k)$ have the form specified in (26) and (33), respectively. As measure of the performance of the proposed estimation algorithm residuals have been calculated, $\boldsymbol{e}(\cdot) = \boldsymbol{n}(\cdot) - \hat{\boldsymbol{n}}(\cdot) = [e_2(\cdot)\ e_3(\cdot)]^T$, and whiteness test is computed. The considered algorithm requires careful selection of the threshold parameters E_1, E_2, E_3 and of parameter M, as defined in (4)–(6), which control the growth characteristics of the network; i.e., if small thresholds are chosen more units are added to the NN. The parameters δ and ξ control the pruning strategy (15); it is important to take into account the system nonstationarity to select these parameters. In other words, slowly dynamic variations imply a bigger δ and a smaller ξ. The parameters γ, p_0, $\boldsymbol{\sigma}_v^2(0)$ and $\sigma_\eta^2(0)$ related to the AEKF algorithm used to update the network parameters of (10) are chosen by trial and error. In the considered experimental tests the numeric values of these parameters are selected as reported in Table I. A sample of the performed estimation tests is given in Figs. 2–4 for the estimation of the uncertainty $\boldsymbol{n}(k)$. In particular, Fig. 2 shows the hidden neurons evolution history for the EMRANAEKF algorithm as it learns sequentially to estimate $\boldsymbol{n}(k)$ from the training data set $(\boldsymbol{\chi}(k), \boldsymbol{n}(k))$, $k = 1, 2, \ldots$. Residuals of the performed

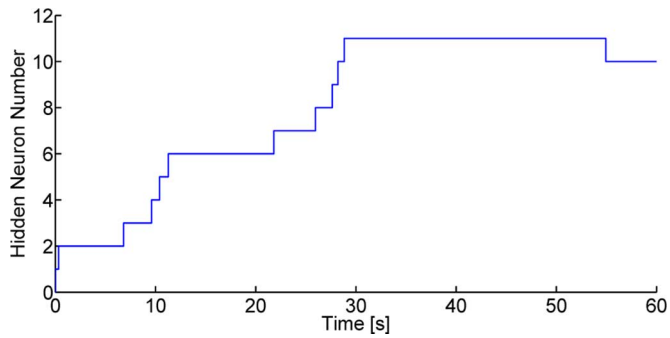


Fig. 2. Evolution of hidden neurons due to growing and pruning for the estimation of $\boldsymbol{n}(k)$; data window for 60 s.

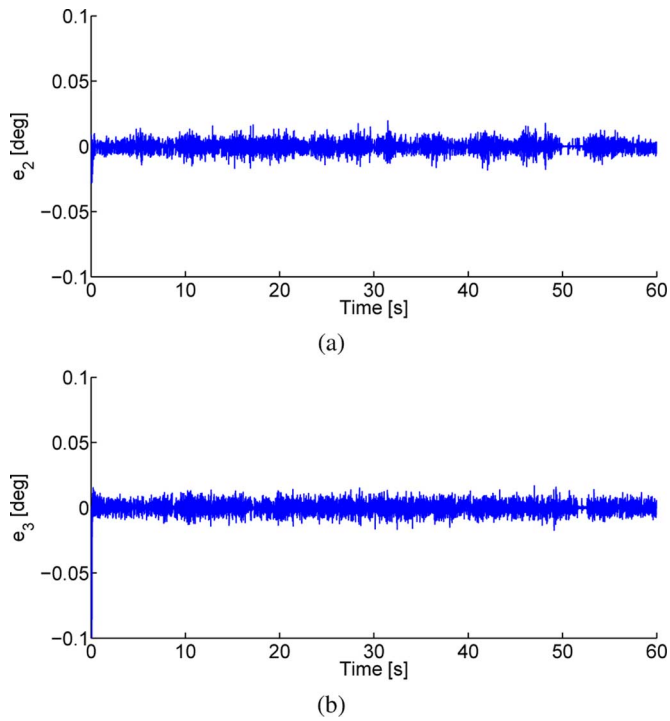


Fig. 3. Residuals obtained by the estimation performed by the network (a) $e_2(\cdot)$; (b) $e_3(\cdot)$.

TABLE I
LEARNING ALGORITHM PARAMETERS

Parameter	Value	Parameter	Value
E_1	$4 \cdot 10^{-4}$	ξ	100
E_2	$3 \cdot 10^{-4}$	γ	0.4
E_3	0.3	p_0	0.2
M	48	$\boldsymbol{\sigma}_v^2(0)$	$[0.001\ 0.001]^T$
δ	$3 \cdot 10^{-3}$	$\sigma_\eta^2(0)$	0.008

estimation shown in Fig. 3(a) and (b), confirm that the implemented NNs are accurate, in particular the mean square of the error (MSE) is $1.27 \cdot 10^{-4}$ and $1.52 \cdot 10^{-4}$ for joint 2 and 3, respectively.

The whiteness test on the estimation errors $e_i(\cdot)$, $i = 2$, 3 (residuals) has been used for network validation [72]. The

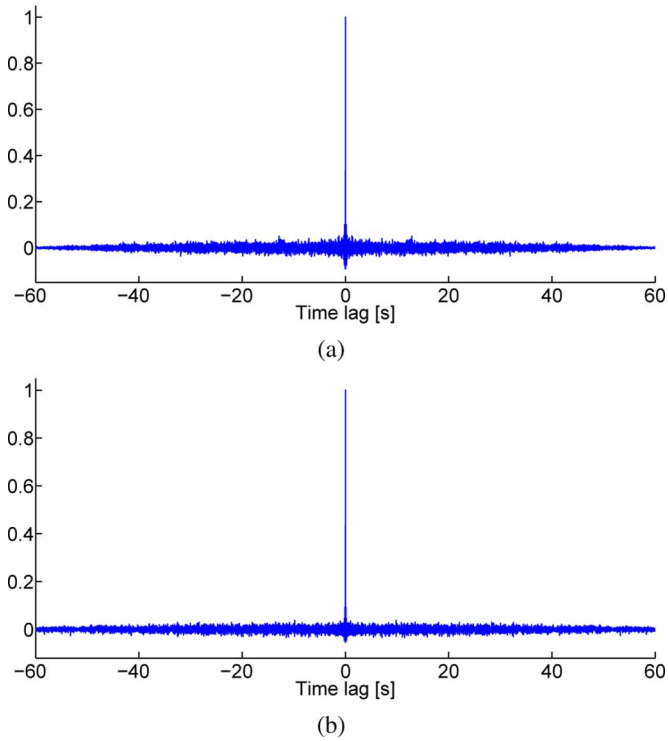


Fig. 4. Sample covariance of the residuals obtained by the estimation performed by the network (a) $e_2(\cdot)$; (b) $e_3(\cdot)$. The whiteness test passes with $\varrho = 0.005$.

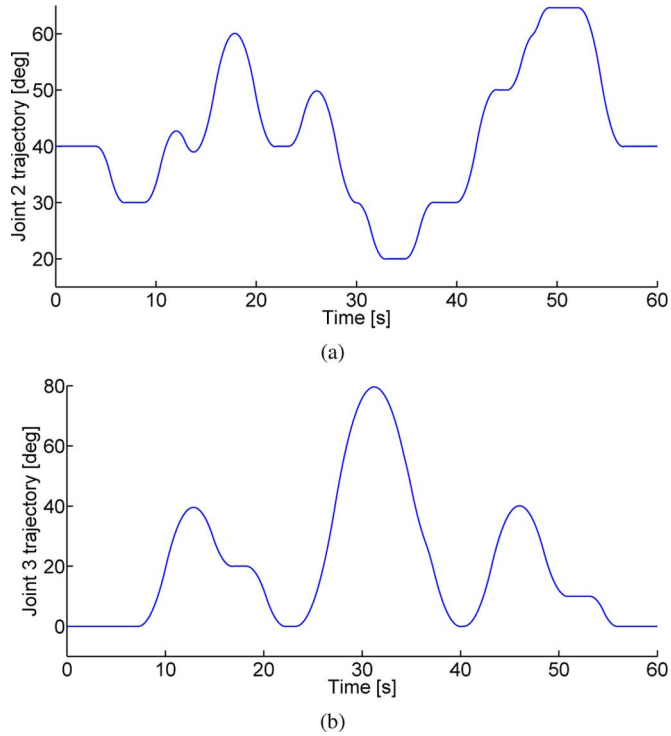


Fig. 5. Reference trajectories used for experiments: (a) joint 2; (b) joint 3.

whiteness of residuals is usually evaluated by computing the sample covariances

$$\hat{R}^K_{e_i}(\tau) = \frac{1}{K}\sum_{k=1}^{K} e_i(k)e_i(k+\tau), i = 2,3 \qquad (38)$$

with $\tau = 1, \ldots, S$. If $e_i(\cdot)$, $i = 2, 3$ are white-noise sequences, then the quantities

$$\zeta_i^{K,S} = \frac{K}{\left(\hat{R}^K_{e_i}(0)\right)^2}\sum_{\tau=1}^{S}\left(\hat{R}^K_{e_i}(\tau)\right)^2, i = 2,3 \qquad (39)$$

will have, asymptotically, a chi-square distribution $\chi^2(S)$ [72]. The independence between residuals can be verified by testing whether $\zeta_i^{K,S} < \chi^2_\varrho(S), i = 2, 3$, the ϱ level of the $\chi^2(S)$-distribution, for a significant choice of ϱ. Typical choices of ϱ range from 0.05 to 0.005. In Fig. 4(a) and (b), the sample covariances of residuals $e_2(\cdot)$ and $e_3(\cdot)$ are reported; the whiteness test passes with $\varrho = 0.005$.

C. Experimental Results

Experimental results have been collected for trajectory tracking tasks performed in the robot joint space. Reference trajectories have been generated by the nonlinear sliding mode discrete-time system reported in [71] that provides smooth trajectories according to user-selectable bounds on trajectories derivatives that can be changed during robot operation. The parameters of the discrete-time SMC law for both joints, are reported in Table II. A set of experimental results is reported in Figs. 6–11, obtained for the robot following the reference trajectories depicted in Fig. 5 (with bounds $|\dot{q}_i| < |35|$ deg/s, $|\ddot{q}_i| < |10|$ deg/s^2, $i = 2, 3$ for joint speed and acceleration, respectively). In these figures, the performance produced by the proposed NN-based SMC are illustrated for the robot following the reference trajectories of Fig. 5 with and without a payload.

TABLE II
SMC PARAMETERS FOR BOTH JOINTS

Param.	Value	Param.	Value
ρ_s	0.0859	θ_{q3}	0.5019
$\min_{\boldsymbol{q}} \|\|\boldsymbol{g}(k) + \Delta\boldsymbol{g}(k)\|\|$	10.3385	$eig(\boldsymbol{\Lambda})$	$[0.92\ 0.91]^T$
θ_{q2}	0.1624	α	0.95

Figs. 6 and 7 show the performance when the robot is without a payload. In particular the tracking errors are displayed in Fig. 6 and the voltage control inputs from the dSPACE controller board are depicted in Fig. 7. Figs. 8 and 9 show the performance when the reference robot motion trajectories are the same as before (see Fig. 5) and the robot moves a payload of 2 Kg; the tracking errors are displayed in Fig. 8 and the voltage control inputs from the dSPACE controller board are depicted in Fig. 9. Comparing with the performance of a robot controller based on a standard discrete-time SMC (without any approximation inside the sector) and based on a PID solution (considering for the robot the same task as in Fig. 5), the proposed NN-based SMC produces smaller tracking errors as reported in Table III. In this table, to summarize the experimental results of Figs. 6–9, the IAE criterion is used, i.e., the integral of the absolute value of the tracking errors

$$I\,AE = \int_0^{T_t} |\pi_i(t)|\,dt \qquad (40)$$

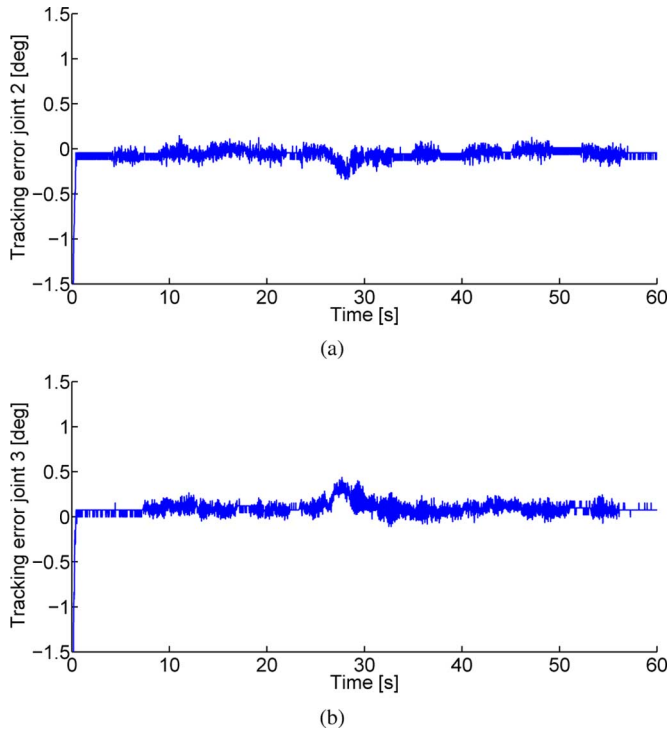


Fig. 6. Results for the robot without a payload—Tracking errors: (a) joint 2; (b) joint 3.

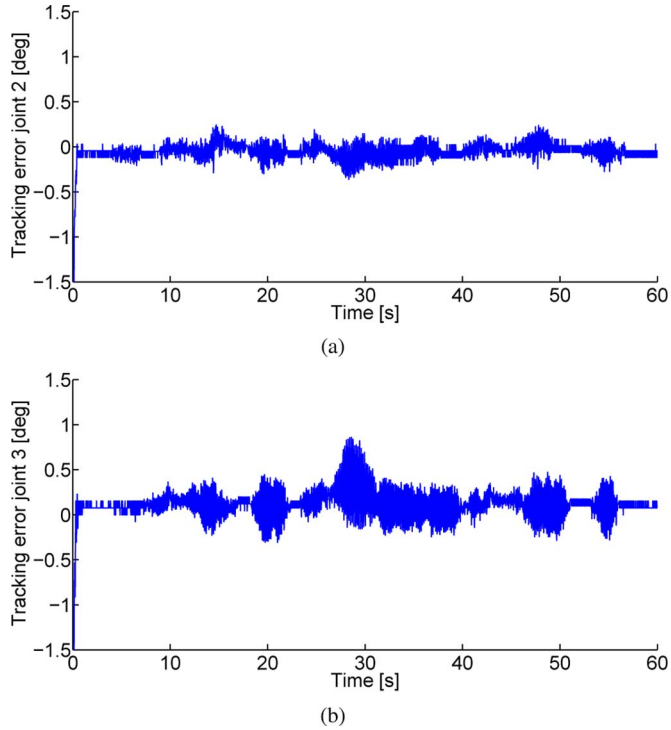


Fig. 8. Results for the robot with a payload—Tracking errors: (a) joint 2; (b) joint 3.

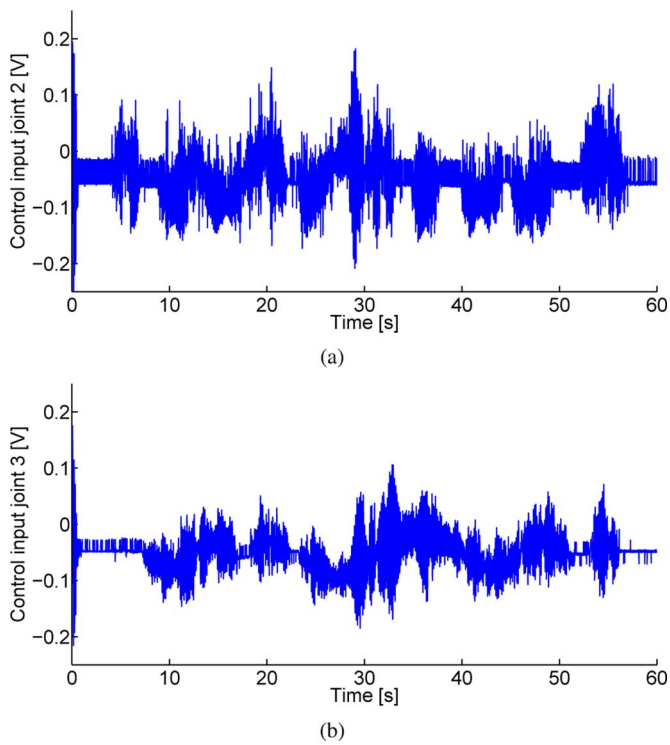


Fig. 7. Results for the robot without a payload—Control inputs: (a) joint 2; (b) joint 3.

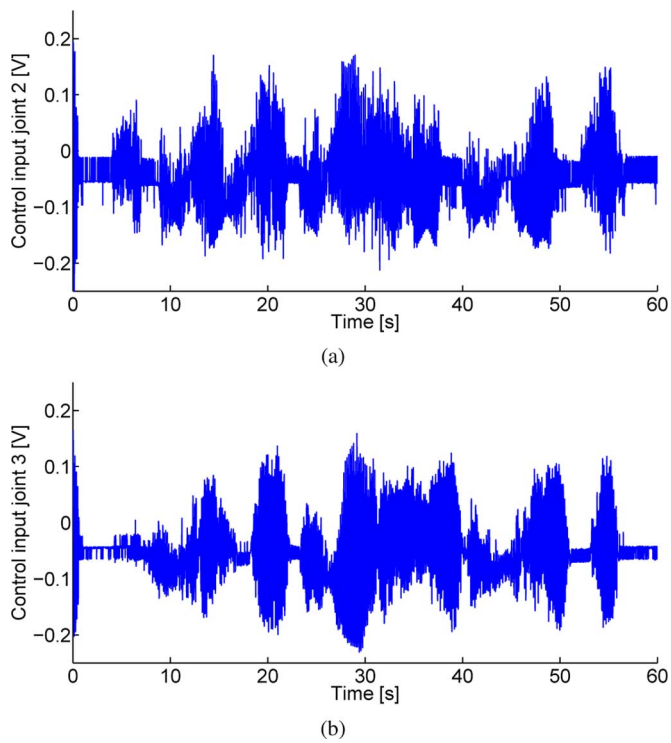


Fig. 9. Results for the robot with a payload—Control inputs: (a) joint 2; (b) joint 3.

where $i = 1, 2$ and T_t is test time. The parameters of the considered PID controller are given in Table IV. Figs. 10 and 11 report the norm of the sliding surfaces $\|\tilde{\boldsymbol{\sigma}}(k)\|$ [see (32)], for the experimental tests of Figs. 6–9. From Figs. 10(a) and 11(a) it is evident that the NN-based SMC causes the sliding surface to decrease and to remain remarkably below the sector threshold of width ρ_s [see (32)], compared with the standard SM controller [see Figs. 10(b) and 11(b)]. Tests with time-varying disturbances affecting the voltage control inputs have been also performed. The reference trajectory is the same as in Fig. 5.

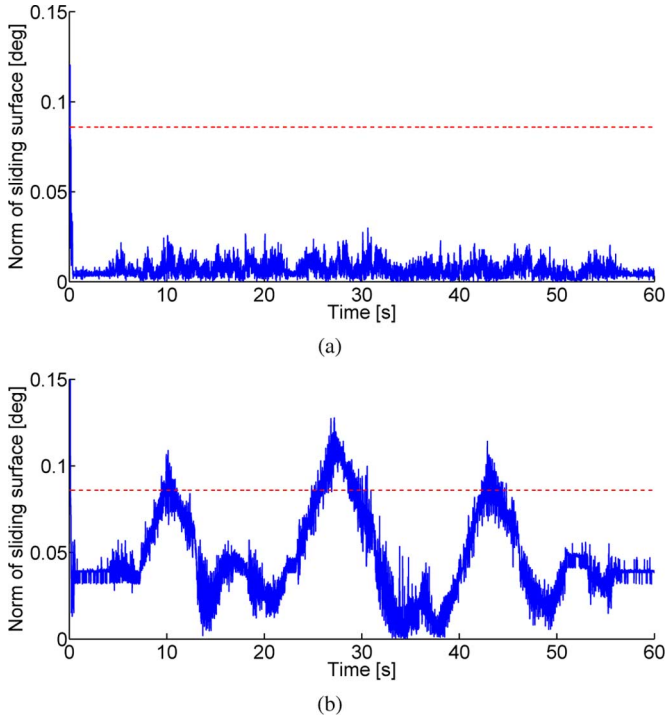


Fig. 10. Results for the robot without a payload—Norm of the sliding surface: red dashed line denotes the threshold ρ_s, blue continuous line denotes the norm of the sliding surface $||\tilde{\boldsymbol{\sigma}}(k)||$ [see (32)]; (a) NN-based SMC; (b) standard SMC.

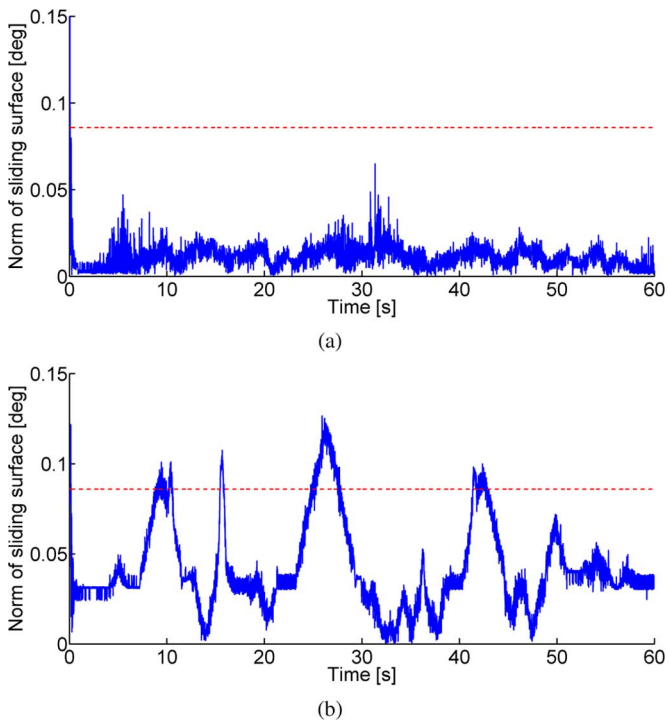


Fig. 11. Results for the robot with a payload—Norm of the sliding surface: red dashed line denotes the threshold ρ_s, blue continuous line denotes the norm of the sliding surface $||\tilde{\boldsymbol{\sigma}}(k)||$ [see (32)]; (a) NN-based SMC; (b) standard SMC.

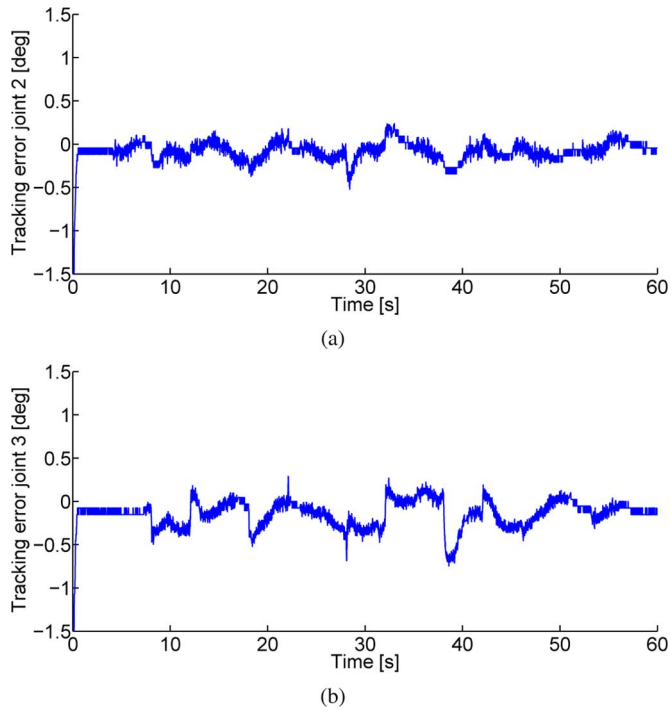


Fig. 12. A time-varying disturbance acts on the voltage control inputs—Tracking errors: (a) joint 2; (b) joint 3.

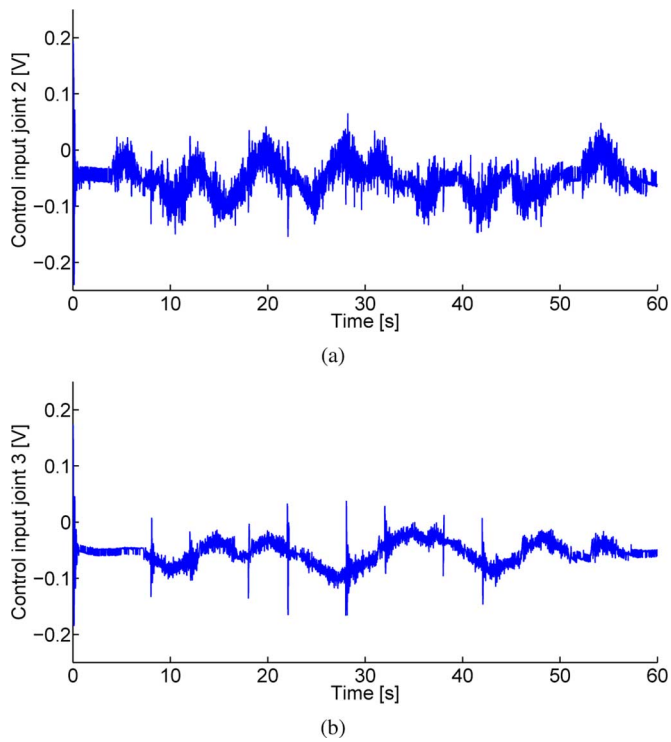


Fig. 13. Time-varying disturbance acts on the voltage control inputs—Control inputs: (a) joint 2; (b) joint 3.

The tracking errors are shown in Fig. 12 and the voltage control inputs from the dSPACE control board are depicted in Fig. 13. Comparing with the performance of a NN-based SM robot controller equipped with a standard EKF (called EKF-EMRAN SMC), i.e., without the online adjustment of the noise statistic, the proposed NN-based SMC equipped with the AEKF (called AEKF-EMRAN SMC) produces smaller tracking errors as reported in Table V; the IAE criterion is used to summarize the experimental results of Figs. 12 and 13. Fig. 14 reports the norm of the sliding surfaces $||\tilde{\boldsymbol{\sigma}}(k)||$ [see (32)], for the experimental

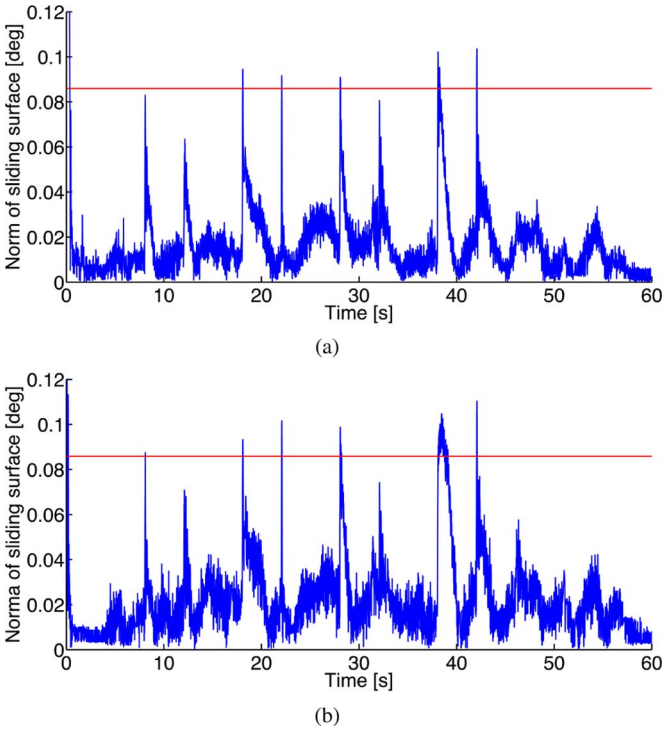


Fig. 14. Results with a time-varying disturbance acting on the voltage control inputs—Norm of the sliding surface: red dashed line denotes the threshold ρ_s, blue continuous line denotes the norm of the sliding surface $\|\tilde{\sigma}(k)\|$ [see (32)]; (a) AEKF-EMRAN SMC; (b) EKF-EMRAN SMC.

TABLE III
PERFORMANCE COMPARISON—FIGS. 6–9

Controllers	IAE (no payload)		IAE (payload)	
	q_2	q_3	q_2	q_3
NN-based SMC	6.57	6.84	7.30	7.94
standard SMC	8.84	9.46	12.30	13.92
PID	14.92	15.99	18.57	21.45

TABLE IV
PID CONTROLLER PARAMETERS

Gain	Value q_2	Value q_3
Proportional	50	30
Integral	15	10
Derivative	1.378	1.425

TABLE V
PERFORMANCE COMPARISON—FIGS. 12 AND 13

Controllers	IAE q_2	IAE q_3
AEKF-EMRAN SMC	8.22	8.95
EKF-MRAN SMC	9.51	10.27

tests of Figs. 12 and 13. From Fig. 14(a) it can be noted that the AEKF-EMRAN SMC produces a smaller sliding surface compared with the EKF-EMRAN SMC [see Fig. 14(b)].

V. CONCLUDING REMARKS

In this paper, a NN-based discrete-time SMC algorithm has been proposed for the control of a planar robotic manipulator. The approximation capability of NNs is used to learn about system uncertainties and it is guaranteed that the tracking error is arbitrarily small in norm. An extended minimal resource allocating network algorithm has been analyzed for the on-line learning of system uncertainties. This algorithm increases the number of RBFN hidden neurons depending on the input-output data and an adaptive extended Kalman filter is used to update all the parameters of the RBFN. The filter adaptation mechanism has been introduced for the adjustment of the noise statistics in order to allow the filter to cope with realistic operating conditions. A pruning strategy is also considered to remove those hidden units which end up to give a contribution to the network output. This permits to have a more parsimonious network topology that leads to a reduction in the prediction time. Moreover, for the real-time implementation the computational load of the algorithm has been reduced by a "winner neuron" strategy. The proposed control law has been tested on a ERICC robot arm. Experimental evidence shows good trajectory tracking performance as well as robustness in the presence of model inaccuracies, disturbances and payload perturbations. The developed controller provided remarkably improved tracking performance both with respect to the standard discrete-time SMC law and to a PID based controller.

REFERENCES

[1] S. Talole, J. Kolhe, and S. Phadke, "Extended-state-observer-based control of flexible-joint system with experimental validation," *IEEE Trans. Ind. Electron.*, vol. 57, no. 4, pp. 1411–1419, Apr. 2010.
[2] C. Abdallah, D. Dawson, P. Dorato, and M. Jamshidi, "Survey of robust control for rigid robots," *IEEE Control Syst. Mag.*, vol. 11, no. 2, pp. 24–30, Feb. 1991.
[3] C. Kuo and S. Wang, "Nonlinear robust industrial robot control," *J. Dyn. Syst., Meas., Contr.*, vol. 111, no. 1, pp. 24–30, 1989.
[4] A. Ibrahimbegovic, C. Knopf-Lenoir, A. Kucerova, and P. Villon, "Optimal design and optimal control of elastic structures undergoing finite rotations," *Int. J. Numer. Meth. Eng.*, vol. 61, no. 14, pp. 2428–2460, 2004.
[5] B. Song and A. Koivo, "Nonlinear predictive control with application to manipulator with flexible forearm," *IEEE Trans. Ind. Electron.*, vol. 46, no. 5, pp. 923–932, Oct. 1999.
[6] T. Shibata and T. Murakami, "Null space motion control by PID control considering passivity in redundant manipulator," *IEEE Trans. Ind. Inf.*, vol. 4, no. 4, pp. 261–270, Nov. 2008.
[7] T. F. Pazelli, M. H. Terra, and A. A. Siqueira, "Experimental investigation on adaptive robust controller designs applied to a free-floating space manipulator," *Contr. Eng. Pract.*, vol. 19, no. 4, pp. 395–408, 2011.
[8] S. Islam and X. Liu, "Robust sliding mode control for robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2444–2453, Jun. 2011.
[9] L. Capisani and A. Ferrara, "Trajectory planning and second order sliding mode motion/interaction control for robot manipulators in unknown environments," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3189–3198, Aug. 2011.
[10] M. Jin, J. Lee, P. H. Chang, and C. Choi, "Practical nonsingular terminal sliding-mode control of robot manipulators for high-accuracy tracking control," *IEEE Trans. Ind. Electron.*, vol. 56, no. 9, pp. 3593–3601, Sep. 2009.
[11] J.-X. Xu, H. Hashimoto, J.-J. Slotine, Y. Arai, and F. Harashima, "Implementation of VSS control to robotic manipulators-smoothing modification," *IEEE Trans. Ind. Electron.*, vol. 36, no. 3, pp. 321–329, Aug. 1989.
[12] L. Capisani, A. Ferrara, and L. Magnani, "Second order sliding mode motion control of rigid robot manipulators," in *Proc. 46th IEEE Conf. Decision Contr.*, New Orleans, LA, 2007, pp. 3691–3696.
[13] F. Sun, L. Li, H.-X. Li, and H. Liu, "Neuro-fuzzy dynamic-inversion-based adaptive control for robotic manipulators—Discrete time case," *IEEE Trans. Ind. Electron.*, vol. 54, no. 3, pp. 1342–1351, Jun. 2007.
[14] V. Utkin, *Sliding Modes in Control and Optimization*. Berlin, Germany: Springer-Verlag , 1992.

[15] A. Zinober, *Variable Structure and Lyapunov Control*. Berlin, Germany: Springer-Verlag, 1994.

[16] A. Sabanovic, "Variable structure systems with sliding modes in motion control—A survey," *IEEE Trans. Ind. Inf.*, vol. 7, no. 2, pp. 212–223, May 2011.

[17] F. Sun, Z. Sun, and G. Feng, "An adaptive fuzzy controller based on sliding mode for robot manipulators," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 5, pp. 661–667, May 1999.

[18] Y. Guo and P. Woo, "An adaptive fuzzy sliding mode controller for robotic manipulators," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 33, no. 2, pp. 149–159, Feb. 2003.

[19] M. Ertugrul and O. Kaynak, "Neuro sliding mode control of robotic manipulators," *Mechatronics*, vol. 10, no. 1–2, pp. 239–263, 2000.

[20] Z. Chen, C. Shan, and H. Zhu, "Adaptive fuzzy sliding mode control algorithm for a nonaffine nonlinear system," *IEEE Trans. Ind. Inf.*, vol. 3, no. 4, pp. 302–311, Nov. 2007.

[21] L. Wang, T. Chai, and L. Zhai, "Neural-network-based terminal sliding-mode control of robotic manipulators including actuator dynamics," *IEEE Trans. Ind. Electron.*, vol. 56, no. 9, pp. 3296–3304, Sep. 2009.

[22] H. Chaoui, P. Sicard, and W. Gueaieb, "ANN-based adaptive control of robotic manipulators with friction and joint elasticity," *IEEE Trans. Ind. Electron.*, vol. 56, no. 8, pp. 3174–3187, Aug. 2009.

[23] H. Hu and P.-Y. Woo, "Fuzzy supervisory sliding-mode and neural-network control for robotic manipulators," *IEEE Trans. Ind. Electron.*, vol. 53, no. 3, pp. 929–940, Jun. 2006.

[24] X. Chen, T. Fukuda, and K. Young, "Adaptive quasi-sliding-mode tracking control for discrete uncertain input-output systems," *IEEE Trans. Ind. Electron.*, vol. 48, no. 1, pp. 216–224, Feb. 2001.

[25] M. Corradini and G. Orlando, "A discrete adaptive variable-structure controller for MIMO systems, and its application to an underwater ROV," *IEEE Trans. Contr. Syst. Technol.*, vol. 5, no. 3, pp. 349–359, May 1997.

[26] K. Furuta, "VSS type self-tuning control," *IEEE Trans. Ind. Electron.*, vol. 40, no. 1, pp. 37–44, Feb. 1993.

[27] P. Lee and J. Oh, "Improvements on VSS-type self-tuning control for a tracking controller," *IEEE Trans. Ind. Electron.*, vol. 45, no. 2, pp. 319–325, Apr. 1998.

[28] O. Kaynak and A. Denker, "Discrete-time sliding mode control in the presence of system uncertainty," *Int. J. Contr.*, vol. 57, no. 5, pp. 1177–1189, 1993.

[29] M. Corradini, G. Ippoliti, S. Longhi, and G. Orlando, "A quasi-sliding mode approach for robust control and speed estimation of PM synchronous motors," *IEEE Trans. Ind. Electron.*, vol. 59, no. 2, pp. 1096–1104, Feb. 2012.

[30] S. Haykin, *Neural Networks: A Comprehensive Foundation (2nd Edition)*. London, U.K.: Prentice Hall, 1999.

[31] G. Acciani, G. Brunetti, and G. Fornarelli, "Application of neural networks in optical inspection and classification of solder joints in surface mount technology," *IEEE Trans. Ind. Inf.*, vol. 2, no. 3, pp. 200–209, Aug. 2006.

[32] T. Orlowska-Kowalska and M. Kaminski, "FPGA implementation of the multilayer neural network for the speed estimation of the two-mass drive system," *IEEE Trans. Ind. Inf.*, vol. 7, no. 3, pp. 436–445, Aug. 2011.

[33] A. Gomperts, A. Ukil, and F. Zurfluh, "Development and implementation of parameterized FPGA-Based general purpose neural networks for online applications," *IEEE Trans. Ind. Inf.*, vol. 7, no. 1, pp. 78–89, Feb. 2011.

[34] M. Efe, "Neural network assisted computationally simple $PI^{\lambda}D^{\mu}$ control of a quadrotor UAV," *IEEE Trans. Ind. Inf.*, vol. 7, no. 2, pp. 354–361, May 2011.

[35] A. Ning, H. Lau, Y. Zhao, and T. Wong, "Fulfillment of retailer demand by using the MDL-Optimal neural network prediction and decision policy," *IEEE Trans. Ind. Inf.*, vol. 5, no. 4, pp. 495–506, Nov. 2009.

[36] S. Simani, "Identification and fault diagnosis of a simulated model of an industrial gas turbine," *IEEE Trans. Ind. Inf.*, vol. 1, no. 3, pp. 202–216, Aug. 2005.

[37] E. Monmasson, L. Idkhajine, M. Cirstea, I. Bahri, A. Tisan, and M. Naouar, "FPGAs in industrial control applications," *IEEE Trans. Ind. Inf.*, vol. 7, no. 2, pp. 224–243, May 2011.

[38] A. Giaquinto, G. Fornarelli, G. Brunetti, and G. Acciani, "A neuro-fuzzy method for the evaluation of soldering global quality index," *IEEE Trans. Ind. Inf.*, vol. 5, no. 1, pp. 56–66, Feb. 2009.

[39] R.-J. Wai and P.-C. Chen, "Robust neural-fuzzy-network control for robot manipulator including actuator dynamics," *IEEE Trans. Ind. Electron.*, vol. 53, no. 4, pp. 1328–1349, Jun. 2006.

[40] C.-S. Chen, "Dynamic structure neural-fuzzy networks for robust adaptive control of robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 55, no. 9, pp. 3402–3414, Sep. 2008.

[41] T. Ozaki, T. Suzuki, T. Furuhashi, S. Okuma, and Y. Uchikawa, "Trajectory control of robotic manipulators using neural networks," *IEEE Trans. Ind. Electron.*, vol. 38, no. 3, pp. 195–202, Jun. 1991.

[42] A. Ishiguro, T. Furuhashi, S. Okuma, and Y. Uchikawa, "A neural network compensator for uncertainties of robotics manipulators," *IEEE Trans. Ind. Electron.*, vol. 39, no. 6, pp. 565–570, Dec. 1992.

[43] F. Lewis, A. Yesildirek, and K. Liu, "Multilayer neural net robot controller: Structure and stability proofs," *IEEE Trans. Neural Netw.*, vol. 7, no. 2, pp. 388–399, Apr. 1996.

[44] M.-C. Chien and A.-C. Huang, "Adaptive control for flexible-joint electrically driven robot with time-varying uncertainties," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 1032–1038, Apr. 2007.

[45] F. Sun, Z. Sun, and P. Woo, "Neural network-based adaptive controller design of robotic manipulators with an observer," *IEEE Trans. Neural Netw.*, vol. 12, no. 1, pp. 54–67, Feb. 2001.

[46] R. M. Sanner and J.-J. E. Slotine, "Stable adaptive control of robot manipulator using neural networks," *Neural Comput.*, vol. 7, no. 3, pp. 753–790, 1995.

[47] S. Ge, T. Lee, and C. Harris, *Adaptive Neural Network Control of Robotic Manipulators*. Singapore: World Scientific, 1998.

[48] F. Lewis, S. Jagannathan, and A. Yeşildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. London, U.K.: Taylor & Francis, 1999.

[49] M. Cavalletti, G. Ippoliti, and S. Longhi, "Lyapunov-based switching control using neural networks for a remotely operated vehicle," *Int. J. Contr.*, vol. 80, no. 7, pp. 1077–1091, 2007.

[50] S. Chen, C. Cowan, and P. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Apr. 1991.

[51] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm," *IEEE Trans. Neural Netw.*, vol. 9, no. 2, pp. 308–318, Apr. 1998.

[52] N. Sundararajan, P. Saratchandran, and Y. Li, *Fully Tuned Radial Basis Function Neural Networks for Flight Control*. London, U.K.: Kluwer, 2002.

[53] J. Platt, "A resource allocating network for function interpolation," *Neural Comput.*, vol. 3, pp. 213–225, 1991.

[54] V. Kadirkamanathan and M. Niranjan, "Function estimation approach to sequential learning with neural network," *Neural Comput.*, vol. 5, pp. 954–975, 1993.

[55] L. Jetto, S. Longhi, and G. Venturini, "Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots," *IEEE Trans. Robot. Automat.*, vol. 15, no. 2, pp. 119–129, Apr. 1999.

[56] A. Giantomassi, G. Ippoliti, S. Longhi, I. Bertini, and S. Pizzuti, "Online steam production prediction for a municipal solid waste incinerator by fully tuned minimal RBF neural networks," *J. Proc. Contr.*, vol. 21, no. 1, pp. 164–172, Jan. 2011.

[57] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, "Neural networks for control systems—A survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.

[58] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, no. 9, pp. 1481–1497, Sep. 1990.

[59] A. Jazwinsky, *Stochastic Processes and Filtering Theory*. New York: Academic Press, 1970.

[60] S. Fioretti and L. Jetto, "Low a priori statistical information model for optimal smoothing and differentiation of noisy signal," *Int. J. Adapt. Contr. Signal Proc.*, vol. 8, pp. 305–320, 1994.

[61] K. Szabat and T. Orlowska-Kowalska, "Performance improvement of industrial drives with mechanical elasticity using nonlinear adaptive Kalman filter," *IEEE Trans. Ind. Electron.*, vol. 55, no. 3, pp. 1075–1084, Mar. 2008.

[62] K. H. Kim, J. G. Lee, and C. G. Park, "Adaptive two-stage extended Kalman filter for a fault-tolerant INS-GPS loosely coupled system," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 45, no. 1, pp. 125–137, Jan. 2009.

[63] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics Modelling, Planning and Control*, ser. Advanced Textbooks in Control and Signal Processing. Berlin, Germany: Springer-Verlag, 2009.

[64] S. Nicosia and P. Tomei, "Robot control by using only joint position measurements," *IEEE Trans. Autom. Contr.*, vol. 35, no. 9, pp. 1058–1061, Sep. 1990.

[65] K. Young, V. Utkin, and U. Ozguner, "A control engineer's guide to sliding mode control," *IEEE Trans. Contr. Syst. Technol.*, vol. 7, pp. 328–342, Jul. 1999.
[66] W. Gao, Y. Wang, and A. Homaifa, "Discrete-time variable structure control systems," *IEEE Trans. Ind. Electron.*, vol. 42, no. 2, pp. 117–122, Apr. 1995.
[67] B. Wang, X. Yu, and G. Chen, "ZOH discretization effect on single-input sliding mode control systems with matched uncertainties," *Automatica*, vol. 45, pp. 118–125, 2009.
[68] B. Wang, X. Yu, and L. Wang, "Convergence accuracy analysis of discretized sliding mode control systems," in *Proc. 11th Int. Conf. Control, Autom., Robot., Vis.*, Singapore, 2010, pp. 1370–1374.
[69] B. Wang, X. Yu, and X. Li, "Zoh discretization effect on higher-order sliding-mode control systems," *IEEE Trans. Ind. Electron.*, vol. 55, no. 11, pp. 4055–4064, Nov. 2008.
[70] 2011 [Online]. Available: http://www.dspaceinc.com
[71] R. Zanasi, C. G. L. Bianco, and A. Tonielli, "Nonlinear filters for the generation of smooth trajectories," *Automatica*, vol. 36, pp. 439–448, 2000.
[72] L. Ljung, *System Identification, Theory for the User*. London, U.K.: Prentice Hall PTR, 1999.

Maria Letizia Corradini (M'98–SM'04) received the Doctor's degree in electronic engineering from the University of Ancona, Ancona, Italy, and the Ph.D. degree from the University of Bologna, Bologna, Italy, in 1992.

She is currently a Full Professor of Automatic Control at the Scuola di Scienze e Tecnologie (Mathematics Division) of the University of Camerino, Camerino, Italy. After having been employed by Honeywell Bull Italia, she was with the Dipartimento di Elettronica e Automatica of the University of Ancona as an Assistant Professor. She became Associate Professor of Automatic Control in 1998 at the University of Lecce, then Full Professor in 2001. She has published more than 150 papers on top rank international journals, books, and refereed conferences in the following areas: variable structure control; switching control; robust control in the presence of nonsmooth nonlinearities in actuators and sensors; fault tolerant control, modeling of biological systems; and mobile and underwater robotics.

Dr. Corradini has been serving as Associate Editor of the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY and of the IEEE Conference Editorial Board since 2004.

Valentino Fossi received the B.S. degree in informatics and automation engineering from the Università Politecnica delle Marche, Ancona, Italy, in 2010. He is currently working toward the M.S. degree in informatics and automation engineering at the same University.

His main interests include nonlinear control, stochastic systems control, and vision systems.

Andrea Giantomassi received the M.S. degree in industrial automation engineering and the Ph.D. degree in engineering science from the Università Politecnica delle Marche, Ancona, Italy, in 2008 and 2012, respectively. His research interests include modeling, identification and control of nonlinear systems, neural-network-based identification and control, signal based fault detection and diagnosis in industrial systems, and power consumption and efficiency optimization in building automation.

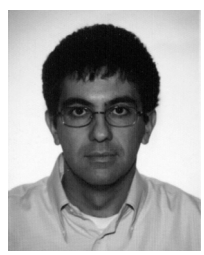
Gianluca Ippoliti (M'12) received the Doctor's degree in electronic engineering and the Ph.D. degree in intelligent artificial systems from the Università Politecnica delle Marche (formerly University of Ancona), Ancona, Italy, in 1996 and 2002, respectively.

From 1997 to 1998, he was with ISERM Unitè 103, Montpellier, France, and then with the University of Montpellier I, Montpellier, France, in the framework of the European research projects CAMARN and MOBINET. From 2002 to 2005, he was a Postdoctoral Fellow at the Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione, Università Politecnica delle Marche. Since March 2005, he has been an Assistant Professor at the Università Politecnica delle Marche. His main research interests include switched systems and supervisory control, neural-network-based system identification and control, modeling identification and control of robotic, marine and electromechanical systems, and mobile robot control and localization.

Sauro Longhi (M'92–SM'04) received his Doctor degree in electronic engineering from the University of Ancona, Italy, in 1979, and the Postgraduate Dipl. degree in automatic control from the University of Rome "La Sapienza," Italy, in 1985.

Since 1983, he has been at the Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione, of the Università Politecnica delle Marche—Ancona. Presently, he holds the position of Full Professor in Control Systems Technologies. His research interests include modeling, identification and control of linear systems, control of mobile base robots and underwater vehicles, service robots for assistive applications, web technology in process control and remote control laboratories, power management in hybrid cars, and cooperative control of autonomous agents. In these fields, he has published more then 180 papers in international journals and conferences. His research has been supported by Italian Ministry of Education and University, the Italian National Research Council, the Italian Space Agency and the European Union. He was the NOC Chair and the IPC Chair of the IFAC Conference on Control Applications in Marine Systems, in 2004 and 2007, respectively. Since April 2007 he has been the general Administrator of the academic spin-off IDEA.

Dr. Longhi is a Member of the Technical Committee on Marine Systems in IFAC.



Giuseppe Orlando was born in Isola del Liri (FR), Italy, in 1966, and lives in Osimo (AN). He received the Doctor degree in electronic engineering and the Ph.D. degree in intelligent artificial systems from the Università Politecnica delle Marche, Italy, in 1992, and 1996, respectively.

From March 1994 to June 1994, he was visiting St. Louis Washington University (USA), with Prof. T.J. Tarn as a tutor. From 1997 to 2000, he was a Postdoctoral Fellow at the Dipartimento di Elettronica ed Automatica, at the Università Politecnica delle Marche. Since October 2000, he has been an Assistant Professor at the Università Politecnica delle Marche. His main research interests include robust control, variable structure control, fault tolerant control, and control of marine systems. He is author of about 90 scientific papers in international journals and conference proceedings.