

A RLWPR Network for Learning the Internal Model of an Anthropomorphic Robot Arm

D. Bacciu*[†], L. Zollo*, E. Guglielmelli*, F. Leoni*, A. Starita[†]

*ARTS Lab, c/o Polo Sant'Anna Valdera, Scuola Superiore Sant'Anna, Pisa, Italy

[†]Dipartimento di Informatica, Università degli Studi di Pisa, Pisa, Italy

Email: {bacciu,loredana,eugenio,leoni}@mail-arts.sssup.it, starita@di.unipi.it

Abstract—Studies of human motor control suggest that humans develop internal models of the arm during the execution of voluntary movements. In particular, the internal model consists of the inverse dynamic model of the musculo-skeletal system and intervenes in the feedforward loop of the motor control system to improve reactivity and stability in rapid movements.

In this paper, an interaction control scheme inspired by biological motor control is resumed, i.e. the coactivation-based compliance control in the joint space [1], and a feedforward module capable of online learning the manipulator inverse dynamics is presented.

A novel recurrent learning paradigm is proposed which derives from an interesting functional equivalence between locally weighted regression networks and Takagi-Sugeno-Kang fuzzy systems. The proposed learning paradigm has been named recurrent locally weighted regression networks and strengthens the computational power of feedforward locally weighted regression networks.

Simulation results are reported to validate the control scheme.

I. INTRODUCTION

Biology can be regarded as a suitable source of inspiration for improving performance of motor control schemes for robot manipulators. In the recent years, in particular, the relationship between robotics and biology has been evolving towards a two-fold interaction: on one hand, robotics can derive inspiration from biology and, on the other hand, robotics can provide biologists with a useful means for validating biological models of motor control and sensory motor coordination.

The work presented in this paper has originated as an attempt of overcoming limitations of standard control strategies by applying biological models to robot manipulators in the interaction with unstructured environments.

Particularly, human motor control is taken into consideration in order to realize a robot motor control ensuring stability in slow as well as fast movements and allowing the execution of unskilled movements in unstructured environments [1].

The control of voluntary movements can be thought of as a combination of kinematic and dynamic transformations of sensory inputs into motor outputs. In this work attention is focused only on dynamic transformations, which, from neurophysiological studies, appear to consist of:

- a sensory to motor transformation, which generates the motor command to reach a desired state;
- a motor to sensory transformation, which generates a sensory prediction in response to a motor command.

The Central Nervous System (CNS) realizes these two kinds of transformations using internal dynamic models of the arm musculo-skeletal system [2]. Forward models learn the causal relationship between the actual state, the motor command and the state resulting from the control action, thus realizing the motor to sensory transformation. On the other hand, inverse internal models learn the sensory to motor relationship.

While the role of forward models in the CNS to skillfully control motion is still controversial, the presence of inverse models appears to be essential for the execution of natural movements [2]. In biological motor control it is inconceivable to think of a feedback control which acts as a unique control loop. The presence of significant delay in the transmission of neural information could cause instability in the execution of rapid movements [1]. Hence, a feedforward action realized by the internal inverse model of the arm musculo-skeletal system seems to act in parallel with feedback by ensuring stability in rapid movements. However feedback control is responsible for unskilled movements and interaction with unknown environments.

Adaptation is another key feature of biological systems, which have to deal with unpredictable changes in dynamic and kinematic conditions of the interaction environment as well as of the musculo-skeletal system (such as growing bones, changes in muscles size and strength). In [2] it is shown how internal models of the musculo-skeletal system are learned by the CNS and evidence is provided of their continuous process of adaptation to changes in the force field exerted by the environment.

In the human arm adaptation to an external force field can be achieved by learning the arm inverse dynamic model and changing it dynamically, depending on the environment. To realize a learning module which mimics adaptation in humans, the definition of a suitable reference signal is needed for training the adaptive module of the arm inverse dynamics. The two main methods proposed in the literature are the Distal Teacher [3] and the Feedback Error Learning (FEL) [4].

The Distal Teacher uses a forward dynamic model (FDM) to generate a prediction of the manipulator behavior, which is used to produce a reference signal for the inverse dynamic model (IDM). In FEL, instead, the IDM is learned by using the output of a simple feedback module as training signal.

More recently, a learning scheme has been proposed

which uses FEL to learn the inverse model of the system while the forward model generates a prediction of the manipulator behavior. The forward model is used to calculate an anticipated feedback control action [5].

The above mentioned control schemes use different learning paradigms to on-line update the parameters which define the dynamic model. In [4], for instance, a three-layered static neural network (NN) is used to implement the adaptive feedforward controller. It is a universal approximator but it does not satisfy the minimum disturbance principle [6]. This means that the optimization procedure minimizes the error only on the current trajectory, but it interferes with the already learned information. Moreover, results of the training procedure are hardly extractable and interpretable in terms of expert knowledge.

Neuro-fuzzy (NF) systems [6] can overcome this lack of transparency thanks to fuzzy logic, which allows interpreting the results in terms of fuzzy rules. Furthermore they satisfy the minimum disturbance principle.

As a counter side NF systems need of prior knowledge about the complexity of the problem to be solved in order to determine the structure of the network (i.e. the number and shape of fuzzy sets and rules). However, as for the NN, constructing algorithms can be used to determine the structure of the network. They can help reduce the expert contribution, although they frequently require a careful choice of the meta parameters (often numerous) which determine the structure.

Recurrent neural networks (RNN) are widely used in the field of system identification because they can treat variable behavior over time. This is the case of learning the dynamic model of a robot manipulator, since it requires to account for the temporal relationships among couples of input/output variables [7]. Thus, static networks cannot be used, while RNN can lead to good performance thanks to the dynamic learning.

Another point to consider in the development of the learning module of robot internal model is that global learning algorithms often neglect the local dimension of learning. This often causes poor performance and reduced network robustness to noisy and irrelevant inputs [8].

In [9] a neural system is presented, which uses the idea of local learning to solve regression problems by incrementally building a superposition of local models. This system shows robustness to interference, can deal with noisy data samples, and can generate the network structure without requiring a priori knowledge.

The work presented here resumes a previous work on biomorph control of an anthropomorphic robot arm in the interaction with unstructured environments [1], and develops a learning paradigm for the robot inverse dynamic model. The coactivation-based compliance control in the joint space in [1] includes the parallel action of a feedback proportional-derivative (PD) control and a feedforward control, based on a mathematical formulation of the robot inverse dynamics. A learning paradigm for the feedforward module is proposed for the following main reasons:

- From a computational viewpoint, using a neural mod-

ule to learn the arm inverse dynamics allows reducing the control computational burden. Contemporary, it does not require knowledge about robot dynamics.

- From the point of view of performance, learning the inverse dynamics instead of mathematically computing it offers a higher level of adaptability to different robotic structures as well as to different environmental conditions. An increased level of task generalization and adaptability is expected.
- From a control viewpoint, accuracy and robustness of classical control schemes are preserved through the action of the PD control in the feedback loop, and an improvement in system reactivity is expected thanks to feedforward.

Thus, the FEL approach in [4] is taken as a starting point for the feedforward module and a novel recurrent extension of Locally Weighted Projection Regression (LWPR) networks [10] is proposed. This allows exploiting robustness of local learning and managing time-variant behavior by means of recurrence. The introduction of recurrence has been guided by a novel interpretation of LWPR in terms of Takagi-Sugeno-Kang (TSK) [11] NF systems. This means that each part of the LWPR network can be interpreted in fuzzy logic and what is known for fuzzy systems can be applied to LWPR and vice versa.

In the next section the coactivation-based compliance control scheme will be briefly introduced, by focusing attention on feedforward modules. Section III outlines the basic features of locally weighted regression networks and describes the fundamentals of the functional equivalence between LWPR and TSK systems. Then, the recurrent extension of LWPR is introduced. The final section of the paper is dedicated to the discussion of the simulation results on a 6-d.o.f. robot manipulator.

II. THE COACTIVATION-BASED COMPLIANCE CONTROL SCHEME

The coactivation-based compliance control scheme in the joint space [1] is briefly introduced here. The control law includes a feedback PD control in the joint space and a feedforward control, based on a mathematical formulation of the robot inverse dynamics. Thus, the control output is defined as

$$\tau = \tau_{FF} + \tau_{FB} \quad (1)$$

where τ_{FF} and τ_{FB} are the feedforward and feedback contributions, respectively.

The feedback motor command is generated by a PD control with gravity compensation defined as

$$\tau_{FB} = K_P(c)\tilde{q} - K_D(c)\dot{\tilde{q}} + g(q) \quad (2)$$

where q is the actual joint configuration and $\tilde{q} = q_d - q$ is the position error in the joint space. Matrices K_P and K_D are the proportional and derivative gains which regulate robot stiffness and viscosity, respectively [1]. The term $g(q)$ is an estimate of the gravitational torques acting on the joints.

The proportional and derivative gains in (2) are adjusted by means of the c factor, that has been named coactivation

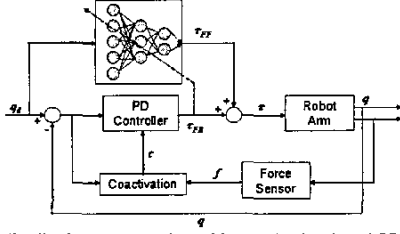


Fig. 1. Feedback Error Learning with coactivation-based PD controller

by analogy with the biological mechanism which regulates viscoelastic behavior of the flexor and extensor muscles.

As regards the feedforward motor action, a learning module is proposed to generate the inverse internal model of the robot arm in a FEL fashion. The dynamic relationship to be learned is

$$\tau_{FF} = B(q_d)\ddot{q}_d + C(q_d, \dot{q}_d)\dot{q}_d + J^T h + F\dot{q}_d \quad (3)$$

where $B \in \mathbb{R}^{n \times n}$ (being n the number of joints) is the joint inertia matrix, $C \in \mathbb{R}^{n \times n}$ accounts for centrifugal and Coriolis forces, $F \in \mathbb{R}^{n \times n}$ is the positive definite matrix of joint viscosity coefficients, J is the Jacobian matrix and the vectors $q_d, \dot{q}_d, \ddot{q}_d \in \mathbb{R}^n$ are the desired joint position, velocity and acceleration, respectively.

The learning module is based on a recurrent version of locally weighted regression (LWR) network, namely the recurrent locally weighted projection regression (RLWPR), whose structure is grown in accordance with a training signal inspired to the FEL scheme (Fig. 1). Hence the robot internal model is developed incrementally without requiring information about the kinematic and dynamic structure of the robot manipulator. The novelty with respect to the work in [9], [10] is the interpretation of LWPR in terms of TSK fuzzy systems, which has suggested the introduction of recurrent connections in the network.

III. THE FEEDFORWARD CONTROL

A. Locally Weighted Regression Networks

Learning inverse dynamics of robot manipulators can be seen as a function approximation, where the input-output relationship is a time dependant, highly non-linear function.

The core of the approach in [9] is the concept of LWR. In this scheme, the estimate of the regression surface for each incoming data point x is obtained by locally fitting a polynomial function of input variables [10]. The LWPR algorithm [10] is a LWR paradigm which approximates non-linear functions by means of piecewise linear local models (LM). In such a scheme the input space is partitioned by gaussian RFs, which define the region of validity of a linear model. The activation strength of the RF is calculated as

$$w_k = \exp\left(-\frac{1}{2}(x - c_k)^T D_k (x - c_k)\right) \quad (4)$$

where $c_k \in \mathbb{R}^n$ is the center of the k -th cell and the distance metric $D_k \in \mathbb{R}^{n \times n}$ is a positive definite matrix which defines shape and dimension of the receptive field. The LM is a linear function of the input variables, thus its output is defined as

$$\hat{y}_k = (x - c_k)^T b_k + b_{0,k} = \tilde{x}^T \beta_k, \quad \tilde{x} = \begin{bmatrix} x - c_k \\ 1 \end{bmatrix} \quad (5)$$

where $\beta_k \in \mathbb{R}^n$ is a vector of linear parameters. Finally, the output of the LWPR network is calculated as a weighted average of all the linear models as

$$\hat{y} = \frac{\sum_{k=1}^K w_k \hat{y}_k}{\sum_{k=1}^K w_k}. \quad (6)$$

As can be seen in (4) and (5), the parameters that have to be learned are the local regression parameters β_k and the RF distance metric D_k , being the center of the cell uniquely determined as soon as the RF is generated.

Thanks to the linearity, the LM parameters can be updated using an online least mean square (LMS) criterion, called Recursive Least Mean Squares (RLMS) [9].

The obtained learning module is the simplest form of LWPR, known in the literature as Receptive Field Weighted Regression (RFWR) [9]. Although it achieves good results in terms of generalization and approximation capabilities [9], it becomes infeasible when the dimension of the input space is larger than (10×1) , since the RLMS has algorithmic complexity $\mathcal{O}(n^2)$ (where n is the input space dimension).

In order to reduce the computational burden of RFWR, the network structure is extended with a preprocessing layer performing a reduction of the input space dimension. Thus, a multivariate regression in a high dimensional space is decomposed into multiple univariate regressions, along selected projection directions which are calculated using Partial Least Squares regression (PLS) [10]. The linear regression included in the PLS procedure allows reducing the algorithmic complexity to $\mathcal{O}(n)$.

Distance metric D_k is learned by means of an incremental stochastic gradient descend algorithm, which uses leave-one-out cross validation and penalty terms to avoid the overfitting problem. The cost function to be minimized can be written as

$$J = \frac{1}{W} \sum_{i=1}^p \frac{w_i \|y_i - \hat{y}_i\|^2}{(1 - w_i z_i^T P_z z_i)^2} + \gamma \sum_{i,j=1}^p D_{ij}^2 \quad (7)$$

where y_i is the reference signal, z_i is the vector of the projected input with inverse covariance matrix P_z , and $\gamma \sum_{i,j=1}^p D_{ij}^2$ represents a penalty term [9].

The update rules are included in an incremental learning system which is capable of on-line generating the structure of the network, namely local models and RF. Projections are also automatically generated on the basis of the desired precision.

B. A Neuro-Fuzzy Interpretation of LWR: the RLWPR network

The recurrent version of locally weighted projection regression (RLWPR) enforces the computational power of LWPR feedforward networks by including recurrent connections in its structure, without breaking the rules of locality.

A novel functional interpretation of LWR in terms of a Takagi-Sugeno-Kang [11] learning system has allowed introducing recurrence in LWPR. A functional equivalence between a TSK learning system and LWPR can be formally

demonstrated by resorting to the same approach used in [12] to provide a fuzzy interpretation of Radial Basis Function networks (RBF). In the following the basic points to demonstrate the equivalence between TSK systems and RFWR networks are presented:

- 1) *Fuzzification*: The RF layer can be compared to the fuzzification layer of a TSK network. In particular, assuming that the number of RFs (indicated with r) is equal to the number of fuzzy rules, the equivalence holds when the fuzzy membership function (MF) is chosen as a Gaussian function with the same center and variance of the RF.
- 2) *Rule firing strength*: The functional equivalence between TSK systems and RFWR networks holds if the T-norm operator used to compute the rule firing strength is the product. In fact, being the distance metric diagonal (or usually chosen diagonal for gaining in computational time), (4) can be written as

$$w_k = \prod_{i=1}^n \exp\left(-\frac{d_{ki}(x_i - c_{ki})^2}{2}\right) \quad (8)$$

which can be interpreted as the premise calculation in a fuzzy system [6].

- 3) *Rule output*: Local models are comparable with the consequences of the first order TSK rules, whose output is calculated as a linear combination of the system inputs. Moreover, being the LM moved to the RF center (see (5)), the equivalence holds when the rule output is biased towards the corresponding MF center c_k . This can be achieved by adding to the rule output the term

$$b_{k-bias} = -\sum_{i=1}^N c_{ki} b_{ki} \quad (9)$$

where b_i ($i = 1, \dots, N$) are the TSK linear parameters.

- 4) *Network output*: Equality holds when the system output is calculated with the same method (e.g. the weighted average in (6)).

These considerations can be easily extended to LWPR. This network, in fact, can be considered as functionally equivalent to a fuzzy system which has a preprocessing layer for feature extraction based on PLS regression. In [13] an application of Fuzzy-PLS to chemometrics can be found. Fewer constraints with respect to RBF-FS equivalence in [12] are required for demonstrating functional equivalence between a TSK system and a LWPR network.

In view of the NF interpretation, each couple RF-LM of a LWPR/RFWR network can be regarded as a fuzzy TSK rule which is trained separately. Inserting local recurrence means, in such a context, adding a term which accounts for the history of the rule activation, thus inserting a feedback connection from the output of each local model to the local model itself (Fig.2). Hence, Eq. 5 can be modified as

$$\hat{y}_k(n+1) = \tilde{x}(n+1)^T \beta_k^n + \hat{y}_k(n)^T \beta_{rec_k}^n \quad (10)$$

to account for the contribution of the feedback connections. The term $\hat{y}_k(n)$ represents the contribution of the feedforward connections, while $\hat{y}_k(n)$ and $\beta_{rec_k}^n$ are the recurrent input and weight vectors, respectively. Comparing

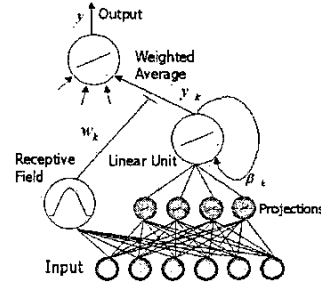


Fig. 2. Recursive Locally Weighted Projection Regression

(10) with (5), it can be noted how the introduction of recurrence enforces the feedforward linear unit activation $\tilde{x}(n+1)^T \beta_k^n$ with a memory term $\hat{y}_k(n)^T \beta_{rec_k}^n$ which accounts for the history of the unit activations.

The learning algorithm for LWPR has been extended, using a RLMS update law, to account for the recurrent connections. Dropping the index k to ease the notation, the learning law for the recurrent weights can be written as

$$\beta_{rec}^{n+1} = \beta_{rec}^n + w P_{rec}^{n+1} y^n e_{cv}^T \quad (11)$$

with

$$P_{rec}^{n+1} = \frac{1}{\lambda_{rec}} \left(P_{rec}^n + \frac{P_{rec}^n y^n y^{nT} P_{rec}^n}{\lambda_{rec} + y^{nT} P_{rec}^n y^n} \right) \quad (12)$$

being the term e_{cv} the cross validation error and λ_{rec} the forgetting factor. It has been included to gradually cancel the contribution from past updates. The learning procedure for the non-recurrent parameters has been adapted to account for the presence of feedback.

In general, LWPR is a fast and stable algorithm thanks to the use of PLS regression, which not only reduces the input space, but also eliminates collinearity in the data. Thus, it ensures a numerically stable learning of the linear parameters.

However, the introduction of recurrence can induce instability in the system, since the feedback connections are not updated in the PLS regression. The covariance matrix P_{rec} can in fact be numerically brittle, typically because of the presence of correlated data in the state vector y^n .

In order to deal with this problem a regularization method is introduced, which takes inspiration from the Expectation Maximization (EM) algorithm proposed in [14]. On the base of the approach in [14], the covariance matrix could be regularized as

$$\tilde{P}_{rec} = P_{rec} + \alpha \Delta^2 I_m \quad (0 < \alpha < 1) \quad (13)$$

$$\Delta^2 = Trace(P_{rec})/m \quad (14)$$

where I_m is an m -dimensional identity matrix. In this way when the data covariance matrix P_{rec} is singular, its zero eigenvalue is replaced by $\alpha \Delta^2$. Hence the regularized matrix \tilde{P}_{rec} becomes regular and stability is always ensured in learning.

In order to apply the regularization scheme to RLWPR, an incremental version of (13) and (14) is defined and included in the update law described by (11) and (12). The resulting online weighted regularization is

$$\tilde{P}_{rec}^n = \tilde{P}_{rec}^n - \frac{\lambda w \tilde{P}_{rec}^n \gamma_i^n \gamma_i^{nT} \tilde{P}_{rec}^n}{1 + \lambda w \gamma_i^{nT} \tilde{P}_{rec}^n \gamma_i^n} \quad (15)$$

with

$$\gamma_i^n = \sqrt{\alpha} \Delta^n \tilde{e}_i \quad (16)$$

where \tilde{e}_i is the i -th unity vector and γ_i^n is the i -th regularization vector.

Equation (15) has the same form of (12), thus the process of regularization can be seen as a standard update of the covariance matrix, where the virtual data vectors $\{\gamma_i^n | i = 1, \dots, m\}$ are included iteratively into \tilde{P}_{rec} . In addition, the regularization can be simplified by exploiting sparsity of the virtual data vectors.

The introduction of recurrence in LWPR enforces the network computational power and improves performance in dynamic learning, as in the case of learning the inverse dynamics of a robot manipulator (that is strongly dependent on time). The recurrent network has a more compact structure and, as it will be shown, does not require taking in input high order terms, such as acceleration, in order to achieve learning [7].

Further, in view of the functional equivalence, what is known for fuzzy systems can be applied to RLWPR and vice-versa. For instance it is possible to extract knowledge from RLWPR networks under the form of a NF rule base, or else, incorporate expert knowledge in the RLWPR structure.

IV. SIMULATION TESTS

The developed control scheme has been tested on a simulated 6-d.o.f. PUMA 560 robot manipulator [15]. A RLWPR network was used to approximate the inverse dynamics of the robot arm. Two different sets of simulation tests have been carried out.

The first set has been aimed at evaluating learning skills of the RLWPR network in comparison with feedforward LWPR network. Therefore, the two paradigms have been implemented to off-line learn the inverse dynamics of the PUMA manipulator [10].

During the tests, consisting of a series of reaching movements, 50.000 data points consisting of joints positions and motor torques have been collected at 100 Hz. The selected reaching movements spanned the whole robot working space, thus providing a significant data set for the robot state space. The training set has been obtained by extracting 45.000 patterns from the collected data, while the remaining 5.000 have been used as test set.

During training, both learning systems received in input the desired joint configuration and provided in output a six-dimensional torque vector that was compared with the reference motor torque, collected during the reaching movements. The dimension of the input space for the LWPR was 18 (being joint position, velocity and acceleration 6×1 vectors, respectively) while the RLWPR input size was 12, being unnecessary to provide the network with joint acceleration. Information about the acceleration was given by the recurrent connections.

In Fig. 3 results of the off-line training are shown. As it can be seen, the normalized RLWPR Mean Squared Error

(nMSE) is below LWPR error, since the beginning of the training. The faster convergence of RLWPR is a consequence of the reduced input dimension and computational burden ensured by the recurrent connections. When the training set is considered, learning converges to a nMSE value of 0.0430 rad for RLWPR and 0.0935 rad for the LWPR network in less than 120000 training presentations. The nMSE on the test set at the end of learning is 0.016 rad for the recurrent network and 0.024 rad for LWPR.

The training phase lasted 1 hour and 15 minutes for the feedforward network by producing 640 RFs. On the other hand, it lasted less than 30 minutes for the recurrent network and generated 381 cells. Thus, the introduction of recurrence produces a reduction of the network size, a faster training phase and higher precision with respect to LWPR. The data on the nMSE error and the network size are reported in Tab. I.

The second set of simulation tests has been carried out to evaluate performance of the developed FEL control scheme in terms of computational burden and capability of compensating for disturbances or unmodelled dynamics. Thus, the recurrent network trained offline in the first session was inserted as a feedforward controller in the FEL architecture. Two versions of the network have been used: the first one is named *RLWPR_{OFF}* and is characterized by frozen weights during control; the second network, named *RLWPR_{FEL}*, can online update its weights like FEL. To do that, the definition of a pseudo-reference signal is required for the RLWPR algorithm to work correctly [16]. The pseudo-target has been defined as

$$\tilde{y}_d(t+1) = \hat{y}(t) + \tau_{FB}(t+1). \quad (17)$$

where $\hat{y}(t)$ is the network output at time t and $\tau_{FB}(t+1)$ is the corresponding feedback compensation at time $t+1$.

In order to have a means of evaluation of the learned inverse dynamic model, a classical feedforward controller based on the Recursive Newton Euler (RNE) method [15] has been taken as a reference.

One of the major drawbacks of classical methods for the computation of the inverse dynamics of a robot manipulator, such as RNE, is the computational burden. As reported in Tab. II, the use of a RLWPR network help reduce the algorithmic complexity of the feedforward controller. The time required by the network to generate the output with on-line learning (i.e. update mechanism) is almost half of the time needed by the classical RNE method to calculate the output. Moreover, generating the network output without on-line learning (i.e. prediction mechanism) takes only 10 msec (in the RNE it is not possible to distinguish between predict and update mechanism because it is based on the mathematical knowledge of robot dynamics). The last row of Tab. II shows the results of a simplified version of RLWPR, that is based on a particular data structure

TABLE I
OFFLINE LEARNING STATISTICS

	nMSE Train [rad]	nMSE Test [rad]	# RFs
LWPR	0.0935	0.024	640
RLWPR	0.0430	0.016	381

TABLE II
PROCESSING TIME FOR PREDICTION AND UPDATE

	Predict [s]	Update [s]	Programming Language
RNE	0.037	0.037	C
RLWPR	0.010	0.020	C++
RLWPR-NV	0.0006	0.001	C++

named neighboring vectors (NV). It exploits the locality of the network to reduce the number of RFs looked up during each operation, thus further reducing the computational burden.

Finally, performance of the control scheme has been evaluated in presence of unmodelled dynamics or disturbances. To this purpose, the same reaching movement has been repeated several times in an environment characterized by dynamical disturbances, generated by a uniform probability distribution at each iteration.

As it can be seen in Fig. 4 and 5, performance of the $RLWPR_{FEL}$ improves over time, while the nMSE for the RNE and RLWPR without online learning follows the stochastic pattern of disturbance. After 50 repetitions of the trajectory the nMSE of $RLWPR_{FEL}$ drops to 0.0461 rad, while $RLWPR_{OFF}$ and RNE achieves nMSE = 0.1596 rad and nMSE = 0.1667 rad, respectively. It is worth noticing in Fig. 5 and 4 that for the first 10 iterations the error in the network with online adaptation is higher than the error in the other two cases. This may be caused by the stochastic nature of the dynamic disturbance and to the use of a pseudo-target training signal in lieu of a target training signal in the FEL learning.

V. CONCLUSION

A bio-inspired control scheme has been described in this paper which combines the coactivation-based compliance control in [1] with an adaptive feedforward module capable of online calculating the inverse dynamics of a robot manipulator.

Particular emphasis has been posed on the learning module used to generate the feedforward control action.

The importance of time dimension in the context of motor learning has been stressed, and a novel recurrent extension of the LWPR network has been proposed to store time-variant information in the feedback connections. Furthermore, it has been shown that the presence of recurrent connections avoids using high order terms in the network, like acceleration, thus reducing network complexity.

The integration of the feedback connections has been inspired by a novel neuro-fuzzy interpretation of LWPR. This allows applying what is known for fuzzy systems to

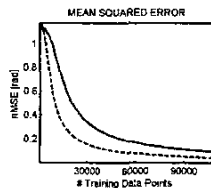


Fig. 3. Normalized Mean Squared Error during training for LWPR (solid) and RLWPR (dashed) networks

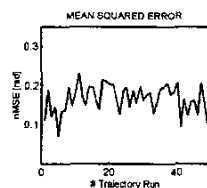


Fig. 4. Normalized Mean Squared Error for each trajectory run in RNE

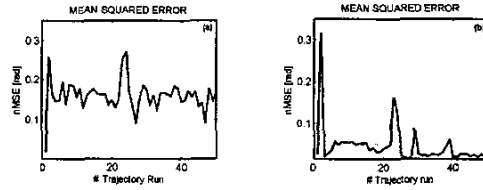


Fig. 5. Normalized Mean Squared Error for each trajectory run in offline taught RLWPR (a) and RLWPR with FEL (b)

LWPR and, in particular, extracting linguistic rules from the structure of the feedforward controller.

It has been shown that the developed learning module can on-line generate its structure and learn to compensate dynamic disturbances without a priori knowledge about the dynamic and kinematic structure of the controlled manipulator.

Simulations tests on a 6-d.o.f. manipulator have been carried out to compare performance of the proposed control scheme with a classical feedforward plus feedback controller. Further, evidence is provided of the improved computational power caused by recurrent connections.

REFERENCES

- [1] L. Zollo, B. Siciliano, E. Guglielmelli, P. Dario, "A bio-inspired approach for regulating visco-elastic properties of a robot arm", *IEEE International Conference on Robotics and Automation*, 2003, Taipei, Taiwan, pp. 3576-3581.
- [2] R. Shadmehr, F.A. Mussa-Ivaldi, "Adaptive representation of dynamics during learning of a motor task", *Journal of Neuroscience*, 1994, vol. 14, pp. 3208-3224.
- [3] M.J. Jordan, D.E. Rumelhart, "Forward models: supervised learning with a distal teacher", *Cognitive Science*, 1992, vol. 16, pp. 307-354.
- [4] M. Kawato, K. Furukawa, R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement", *Biological Cybernetics*, 1987, vol. 57, pp. 169-185.
- [5] M. Katayama, "A neural control model using predictive adjustment mechanism of viscoelastic property of human arm", *Icann Aug. 21-25 2001, Vienna, Austria*, Springer-Verlag.
- [6] J.-S. R. Jang, C.-T. Sun, "Neuro-fuzzy modeling and control", *Proceedings of the IEEE*, 1995, vol. 83, pp. 378-406.
- [7] Y.C. Wang, C.J. Chien, C.C. Teng, "Takagi-Sugeno recurrent fuzzy neural networks for identification and control of dynamic systems", *IEEE International Fuzzy Systems Conference*, 2001, pp. 537 - 540.
- [8] J. Yen, L. Wang, W. Gillespie, "A global-local learning algorithm for identifying Takagi-Sugeno-Kang fuzzy models", *IEEE International Conference on Computational Intelligence*, 1998, pp. 967-972.
- [9] S. Schaal, C.G. Atkeson, "Constructive incremental learning from only local information", *Neural Computation*, 1998, vol. 10, pp. 2047-2084.
- [10] S. Vijayakumar, S. Schaal, "Locally weighted projection regression: an O(n) algorithm for incremental real time learning in high dimensional space", *International Conference on Machine Learning*, 2000, pp. 1079-1086.
- [11] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its applications to modelling and control", *IEEE Transactions on Systems, Man and Cybernetics*, 1985, vol. 15, pp. 116-132.
- [12] J.-S. Roger Jang, C.T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems", *IEEE Transactions on Neural Networks*, 1993, vol. 4, pp. 156 - 159.
- [13] Y.H. Bang, C.K. Yoo, L.-B. Lee, "Nonlinear PLS modeling with fuzzy inference system", *Chemometrics and Intelligent Laboratory Systems*, 2002, vol. 64, pp. 137-155.
- [14] M. Sato, S. Ishii, "On-line EM algorithm for the Normalized Gaussian Network", *Neural Computation*, 2000, vol. 12, pp. 407-432
- [15] P.I. Corke, "A Robotics Toolbox for MATLAB", *IEEE Robotics and Automation Magazine*, 1996, vol. 3, pp. 24-32.
- [16] T. Shibata T., S. Schaal, "Fast learning og biomimetic oculomotor control with nonparametric regression networks", *IEEE International Conference on Robotics and Automation*, San Francisco, 2000, pp. 3847-3854.