Biol Cybern (2012) 106:507–522
DOI 10.1007/s00422-012-0515-5

ORIGINAL PAPER

Biological Cybernetics

Bio-inspired adaptive feedback error learning architecture for motor control

Silvia Tolu · Mauricio Vanegas · Niceto R. Luque · Jesús A. Garrido · Eduardo Ros

Received: 8 November 2010 / Accepted: 31 July 2012 / Published online: 21 August 2012

Abstract This study proposes an adaptive control architecture based on an accurate regression method called Locally Weighted Projection Regression (LWPR) and on a bio-inspired module, such as a cerebellar-like engine. This hybrid architecture takes full advantage of the machine learning module (LWPR kernel) to abstract an optimized representation of the sensorimotor space while the cerebellar component integrates this to generate corrective terms in the framework of a control task. Furthermore, we illustrate how the use of a simple adaptive error feedback term allows to use the proposed architecture even in the absence of an accurate analytic reference model. The presented approach achieves an accurate control with low gain corrective terms (for compliant control schemes). We evaluate the contribution of the different components of the proposed scheme comparing the obtained performance with alternative approaches. Then, we show that the presented architecture can be used for accurate manipulation of different objects when their physical properties are not directly known by the controller. We evaluate how the scheme scales for simulated plants of high Degrees of Freedom (7-DOFs).

S. Tolu (✉) · N. R. Luque · E. Ros
CITIC-Department of Computer Architecture and Technology, ETSI Informática y de Telecomunicación, University of Granada, Granada, Spain
e-mail: stolu@atc.ugr.es

J. A. Garrido
Consorzio Interuniversitario per le Scienze Fisiche della Materia (CNISM), Via Bassi 6, 27100 Pavia, Italy

M. Vanegas
PSPC-group, Department of Informatics, Bioengineering, Robotics and Systems Engineering (DIBRIS), University of Genoa, Genoa, Italy

Keywords Adaptive filter · Feedforward scheme · Cerebellum · Motor control · Machine learning · Internal model

1 Introduction

The problem of controlling a robot of many Degrees of Freedoms (DOFs) is that of determining the forces or torques to be developed by the joint actuators to obtain the right execution of the commanded task (Van der Smagt et al. 1996). Traditional methods are no longer suitable for controlling the complex dynamics of the new generation of light-weight robots (Hirzinger et al. 2000; German Aerospace Center 2011) as the movement is influenced by the state variables of all the joints and the control becomes very complex and highly nonlinear (Van der Smagt 1998). Nonlinearities can dominate the robot dynamics and the feedback gains have to be increased to compensate the resulting tracking error (Gomi and Kawato 1992) for accurately following a predefined desired trajectory. This is dangerous for the system stability and implies noncompliant movements. Furthermore, high gains are unacceptable in autonomous and biological systems as they introduce destabilizing components provided the inherent feedback sensorimotor delays (Porrill and Dean 2007). Therefore, classic feedback control seems to be inappropriate because high gains result in large forces generating potentially dangerous noncompliant movements (Nguyen-Tuong and Peters 2008), making the robot less safe for the environment, mainly in the framework of human-interaction applications, and compromising the closed-loop stability (Jordan 1996). The major drawback of feedback error learning according to Porrill and Dean (2007) is that it requires complex reference structures

for generic redundant and nonlinear systems. Some authors attempted to avoid this problem using high gains in the feedback loop (Miyamura and Kimura 2002). Otherwise, (Gomi and Kawato 1992) described a conventional feedback controller, Proportional Derivative and Acceleration (PDA), for a simple linear case, as an inverse reference model to convert the trajectory error into motor error. Moreover, an analytic computation of the dynamics is complex and in the case of a large number of DOFs, precise dynamics parameters may be unknown. In this case, adaptive models are required for an accurate and stable control during manipulation. Here, we address this problem using a module called Learning Feedback (LF) controller as an inverse reference model. This improves the system behavior and self-adapts by a learning rule through consecutive iterations of the same trajectory.

The cerebellum plays an important role in accurate motor learning, motor adaptation, and cognition (Ito 2000), e.g., computing the inverse dynamics of a body component (Wolpert 1997; Wolpert et al. 1998), delivering feedforward (Kawato 1990; Gomi and Kawato 1992) and feedback terms to the crude control commands from the motor cortex. Recent research studies (Dean et al. 2010) describe the cerebellum as a set of adaptive modules (cerebellar microcomplexes) encoded in the motor control system to improve coordinated movements over time. There are even researches that specifically study how spiking cerebellar-like neural structures can efficiently contribute in control tasks within biologically plausible control schemes (Carrillo et al. 2008; Luque et al. 2011a,c,b). This study presents an architecture in which the cerebellar cortex is embedded in a feedforward loop and the basic cerebellar microcircuit is based on the Marr and Albus' model (Marr 1969; Albus 1971). Inspired on this model, Albus (1975) proposed the CMAC controller capable of learning and retrieving the motor behaviour for joint control. Feedback from sensors in the joints drives the CMAC together with an input command that carries information about the goal or task to be performed. More recently, also Ito (2008) stated that a feedback controller generates a command in the motor cortex that drives the controlled body part accordingly to the desired instruction. The cerebellum forms and adjusts internal models that reproduce the dynamics of the robot body through a learning process as the movement is repeated (Ito 2008). Then, the body inherent characteristics are captured in an internal model in the cerebellum to precisely perform the control of the robot arm by referring to it. This means that the feedback control is replaced by the internal model that reproduces the dynamics of the robot arm. Once the internal model is learned, it helps the brain to perform the task precisely without referring to feedback. The outcome is that the desired motions are predicted, and only few correction forces are required, thus increasing the system's control compliance. Kawato (1990) and Wolpert et al. (1998) proposed an architecture based on feedback error learning (FEL) emulating the role of the cerebellar microcircuit.

A simple cerebellar model can be seen as a single neural network layer (Porrill et al. 2004) (Fig. 1b) where the mossy fibers (MFs) deliver signals that are distributed over many granule cells (GCs) whose outputs are the parallel fibers (PFs). The information sent through the PFs arrives to the Purkinje cells (PCs). The PC has also another input called climbing fiber (CF) which is interpreted as a teaching signal in the Marr and Albus' models or in other terms, it is called the motor error that helps one to adjust the synaptic weights using the covariance learning rule proposed by Sejnowski (1977). CFs enable the cerebellum to form and update internal models (based on these error-related estimates). The motor error is the difference between the desired and the actual motor commands. However, the correct motor command is typically unknown; only sensory errors are available, and how to use this information for motor learning causes the so-called *distal error problem* (Dean et al. 2010). The LF controller generates adaptive feedback commands from the sensory errors avoiding classic PID with high gains and complex reference structures (Porrill and Dean 2007), thus addressing efficiently the motor error problem.

It is important to remark that recent studies highlight the importance of other elements such as interneurons for learning consolidation (Wulff et al. 2009). As input to the Purkinje layer, we use machine learning modules (LWPR kernels) whose adaptive input receptive fields (RFs) can be seen as an abstraction of the granular and molecular layer modules (including also interneurons (Wulff et al. 2009)) that efficiently and accurately deliver clean signals to the Purkinje layer. In our approach, the cerebellar module (C) includes only short term adaptation while consolidation of learned primitives takes place at the machine learning module (LWPR) as also indicated in the results and conclusions sections. There have been some applications of cerebellar models to the control of robot manipulators, all in simple systems such as Gomi and Kawato (1992), Porrill and Dean (2007), Haith and Vijayakumar (2009) and in real robot systems such as Shibata and Schaal (2001).

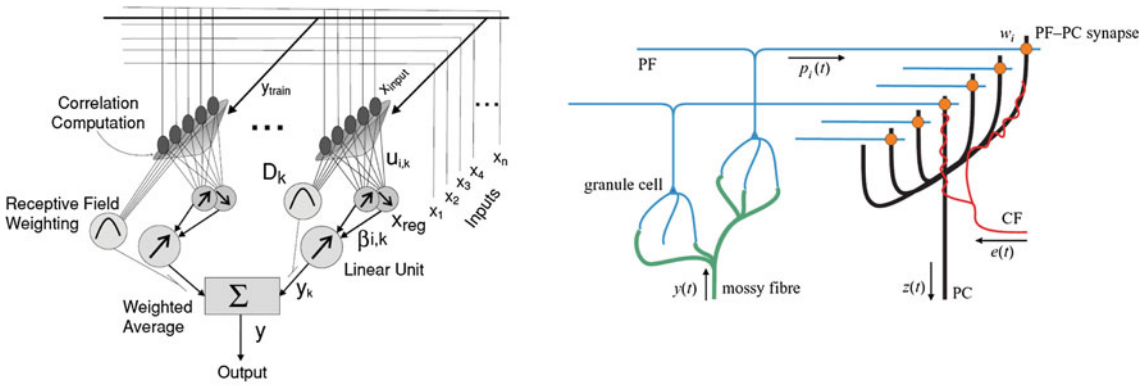
Recent approaches treat the solution of inverse dynamics (also referred to as inverse internal model) as a function approximation problem (Nguyen-Tuong and Peters 2008; Lonini et al. 2009). A robot arm produces a vast amount of data (joint angles, velocities, accelerations, and torques) during its movements, which can be used as training data for the LWPR algorithm (Vijayakumar and Schaal 2000; Vijayakumar et al. 2005). In this sense, machine learning algorithms and the robot systems may help us to develop an understanding of how motor learning takes place in biological systems. As human motor skills are adaptive to changes in the body's physical morphology and the nature of the task being performed, the biological system is able to accomplish

compliant and precise motion. Function approximation can solve the lack of a perfect analitycal model; however, the learned dynamics function represents only a part of the dynamic robot model. In fact, there are some difficulties to know the exact model in the case of miscalibrations of the joints, changes of context, objects under manipulation, etc; thus, the learned function will not be able to prevent all the uncertainties. For this reason, the task of the cerebellar microcircuit is to compensate for these changes while the LWPR incrementally learns the adapted dynamic models. The learned function which emulates the inverse dynamics model of the arm together with the feedback learning module make the robot arm capable of performing movements that are both precise and compliant at the same time and also adaptable to changing situations. It is well-known that the human motor system is able to generate accurate control commands under different environment changing conditions. Wolpert et al. (1998) proposed a modular organized structure of internal models which can be forward and inverse models. The first type predicts the consequences of actions under different contexts while the second one provides commands to achieve desired trajectories. In other words, inverse internal models store a map of the motor apparatus in terms of an inverse dynamics model with a state-space representation. Accordingly, this inverse dynamics model will provide precise command torques over the input state-space and new trajectories are efficiently controlled based on previously learned primitives (Kawato 1999). As a matter of fact, we have done a generalization experiment to evaluate the functional structure of our LWPR internal model. After learning the trajectories defined in Eqs. (16) and (18), LWPR predicts accurate torques when the robot arm has to follow a trajectory given by the summation of previous learned trajectories keeping the desired performance. In fact, the motor commands are predicted by a map from the state-space input composed of positions, velocities and accelerations of desired trajectory, and positions and velocities of the computed trajectory. So, if the new trajectory is closed to the known state-space, the generalization of previous learning will work better. In practice, LWPR generalization performance (that supports our approach's generalization capability) has already been evaluated in Schaal et al. (2002).

Among the global nonlinear function approximators, such as Gaussian Process Regression (GPR) (Williams and Rasmussen 1996), or Support Vector Regression (SVR) (Smola and Schölkopf 2004), LWPR has been successfully used for online incremental learning in robotic platforms (Schaal et al. 2002; Nguyen-Tuong and Peters 2008; Vijayakumar et al. 2005; Atkeson et al. 2000) as it spatially exploits localized linear models to approximate nonlinear functions at a low computational cost. Therefore, the evaluation of the prediction value is quite fast, allowing real-time learning. Besides, the incremental training allows the acquisition and retention of different tasks without interferences among them (Lonini et al. 2009). Figure 1a summarizes the basic process of LWPR learning.

Bearing in mind Fig. 1, we exploit the similarity between the LWPR learning mechanism and the cerebellar circuitry to fuse their functionalities and take advantage both of the potential of the machine learning algorithm and of the cerebellum's role to make fine adjustments to the way an action is performed. In the LWPR, the RF weighting kernels encode the input space like the cerebellar GCs that expansively encode the information coming from the MFs. Previous simulation studies have widely developed the theory of the cerebellar granular layer as a liquid state machine, where the PFs generate a finite but very long sequence of active neuron populations without recurrence (Yamazaki and Tanaka 2007). In a similar way, RF weighting kernels could adapt their weights to select different outputs depending on the current state of the robot arm. Schweighofer et al. (2001) hypothesized that the cerebellar learning is facilitated by a GC sparse code, i.e., a neural code in which the ratio of active neurons is low at any time. Porrill and Dean (2007) stated that both accuracy and learning speed could greatly improve by optimizing the choice of the centers and transforming to an optimal basis of RFs. According to these hypotheses, we exploited the LWPR capabilities to emulate the granular layer with a limited number of resources. LWPR places and adapts efficiently its internal kernels to better represent the input-space with a limited number of them. In fact, unlike the cerebellum, LWPR automatically evaluates the required number of local correlation modules to optimize the network size by incremental learning. In this sense, each LWPR module and its associated RF weights can be seen as providing the firing rate of a PF, while the set of active RF weights can be seen as the current state of the granular layer-processing module (Yamazaki and Tanaka 2007). This state would be propagated through PFs and interneurons to produce more accurate signals at the PCs (Wulff et al. 2009). The major strength of the LWPR is the use of incremental calculation methods during the training, and therefore, it does not require the input data to be stored. Furthermore, the algorithm can cope with highly redundant and irrelevant data input dimensions without any prior knowledge of the data, because it uses an incremental version of the Partial Least Squares regression (PLS).

The major contribution of the presented model is that it manages to learn different nonlinear dynamics with a hybrid approach that uses a machine learning engine (LWPR) and a bio-inspired module (cerebellar-like network). In other words, we exploit the RF weighting of each local model in the LWPR as granular and molecular layer microzones (complexes) in the cerebellum approach. Therefore, the cerebellum module instead of receiving inputs by means of MFs, receives preprocessed signals from the LWPR RFs. This takes advantage of the optimized engine for a compact sensorimotor



(a) The LWPR processing unit (Reproduced from Vijayakumar et al (2005)).

(b) The microcircuit of the cerebellum (Reproduced from Porrill et al (2004)).

Fig. 1 Parallelism between the LWPR processing unit and the cerebellum microcircuit

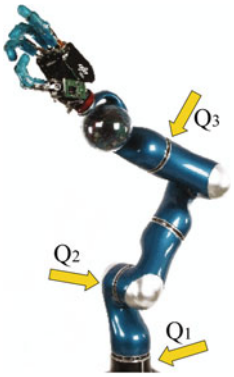


Fig. 2 Light Weight Robot (LWR) arm and hand consisting of seven revolute joints. The three joints used in our simulated 3DOF experiments are explicitly indicated

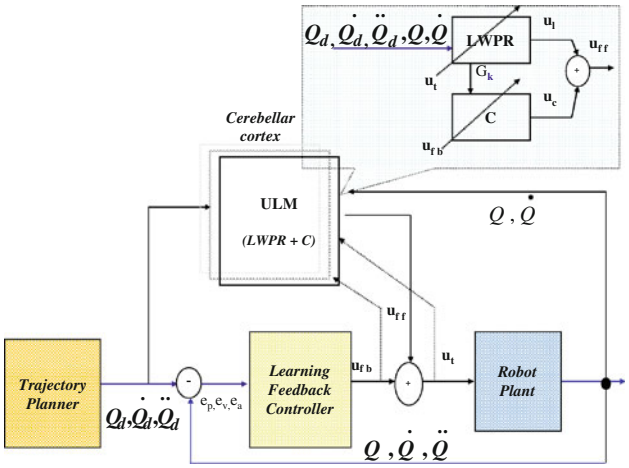


Fig. 3 Block diagram for the Adaptive Feedback Error Learning (AFEL) scheme

representation provided by the LWPR. The LWPR incrementally learns and stores the inverse internal model of the robot arm, while the C module allows a faster control and a more precise movement (Schweighofer et al. 2001). Furthermore, this article studies how the C module infers corrective terms when a noise (related to the inherent noise of the muscle spindle signal and stochastic cells) is introduced in the MFs (Schweighofer et al. 1998) or in other terms in the LWPR inputs.

In the following paragraphs, we will present the advantages of the control architecture system. First, each block of the proposed architecture is presented relating it to the learning rule of the LF controller and the connection between the cerebellum and the LWPR algorithm. Second, we will demonstrate the validity and efficiency of the model with experiments on a 3-DOF and 7-DOF simulated Light Weight Robot (LWR) arm (see Fig. 2). This is the third generation of a 7-DOF robot arm designed by the Institute of Robotics and Mechatronics at the German Aerospace Centre (DLR) (Hirzinger et al. 2000; German Aerospace Center 2011).

2 Control architecture

In this section, the Adaptive Feedback Error Learning (AFEL) architecture, shown in Fig. 3, is presented. It consists of the LF controller which generates the u_{fb} feedback joint torques and the Unit Learning Machine (ULM) which provides the u_{ff} feedforward joint torques. This u_{ff} term is a combination of an u_l prediction term from the LWPR and an u_c prediction term from the cerebellum model. The trajectory planner block computes the desired joint angles, velocities, and accelerations (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$) by inverse kinematics.

The field of nonlinear control theory is very large, therefore, we will focus our attention on a particular method called feedforward nonlinear control (Craig 2005). Considering the analytic robot model, we calculate the joint torques required for a particular trajectory using the following dynamic Eq. (1) of the robot:

$$\tau = M(Q)\ddot{Q} + V(Q, \dot{Q}) + G(Q) + F(Q, \dot{Q}), \quad (1)$$

where $M(Q)$ is the inertia matrix of the manipulator, $V(Q, \dot{Q})$ represents the centrifugal and Coriolis terms, $G(Q)$ is the gravity term, $F(Q, \dot{Q})$ is the model of friction, and Q, $\dot{Q}$, and $\ddot{Q}$ are, respectively, the obtained joint angles, velocities, and accelerations of the robot arm.

In the architecture, the u_t global torque (2) is the summation of the u_{ff} predicted motor command which comes out of the ULM and allows the robot to follow a desired trajectory (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$), and the u_{fb} motor feedback command, generated by the LF controller, which ensures the stability of the trajectory.

$$u_t = u_{ff} + u_{fb}. \quad (2)$$

If the adaptive model is accurate, then the resulting u_{ff} feedforward term will cancel the robot nonlinearities. However, if the inverse dynamic model is not exact, there will be an error between the desired signal and the output of the controlled robot arm (Q, $\dot{Q}$, $\ddot{Q}$). This is also called feedback error and the LF controller output will reflect it. Then, the cerebellum receives the signal which will activate the process of learning (Ito 2008).

As Ito (2008) stated, the cerebellum is composed of many modules called microcomplexes, each of which is a ULM made up of structured neuronal circuits and it encodes an internal model. The input–output relationship of each ULM is adaptively modified by the CFs that convey the error signal. So, the dynamics of the robot are encoded in the ULM which carries out the role of an internal model, as illustrated in Fig. 3. Each microcomplex adapts the corrections for any possible miscalibration, e.g., on account of interaction torques, by a teaching inverse reference signal or appropriate motor command (u_{fb}). The latter is learned by the LF controller on consecutive iterations of the task and also assures the stability of the system without using high gains. Once the internal model is learned, the ULM performs the movement precisely with a low contribution from the feedback. Inside the ULM block, the LWPR algorithm plays the important role of internal model, or in other words, it learns the inverse model of the robot arm, while corrections are applied following the trajectory. In addition to this, the ULM also consists of a set of uniform cerebellar circuits which are capable of learning the input–output relationship of dynamic processes by means of the long-term depression induced in the synapses between PFs and PCs. The error signal conveyed by the CF adaptively modifies this relationship. Analogously, this plastic site is represented by the C module—in function rather than form—in our system, and it is quite sensitive to the representation of the input space (Porrill and Dean 2007; Schweighofer et al. 2001). The error signal computed by the LF and sent to the C module is the effect of the system action that is minimized through the Hebbian rule (13) by analogy with the *distal teacher* problem formulated by Jordan and Rumelhart (1992). Further details about the ULM are given in Sect. 2.2.

Summarizing, the cerebellum leads the model abstraction engine (LWPR), captures through optimized representation the sensorimotor complexes and produces u_c corresponding torques to reduce the u_{fb} teaching signal to the least possible amount (error related estimate). The LWPR engine incrementally learns from the u_t global torques by abstracting the whole model.

2.1 Learning feedback controller

The LF controller overcomes the lack of a precise robot arm dynamic model ensures the stability of the system, and enables the control architecture improve its movement performance. This is achieved by adding a feedback control torque to the one which is provided by the known part of the model. The $\hat{u}_{fb_r}$ feedback torque, shown in the following Eq. (3), is adjusted through a learning rule after consecutive repetitions of the same task, $r = 0, 1, \ldots$ (where r indicates the iteration number).

The dynamics of the robot can be written as

$$\tau = \hat{M}(Q_d)\ddot{Q}_d + \hat{V}(Q_d, \dot{Q}_d) + \hat{G}(Q_d) + \hat{u}_{fb_r}. \quad (3)$$

Keeping in mind the dynamic model described in Eq. (1), considering a nonmodeled $\hat{u}_{fb_r}$ friction term to be added to the estimated terms, and substituting Eq. (3), we obtain error Eqs. (4) and (5) of the closed loop of the control system in Fig. 3:

$$\ddot{e} = M^{-1}(F - \hat{u}_{fb_r}) \quad (4)$$

or in a more compact form:

$$\ddot{e} = (B - B_r), \quad (5)$$

where $\ddot{e} = \ddot{Q}_d - \ddot{Q}$, $B = M^{-1}F$, $B_r = M^{-1}\hat{u}_{fb_r}$. For every joint, the Eq. (5) becomes

$$\ddot{e}_i = (B_i - B_{ir}). \quad (6)$$

In the last Expression (6), term B_i is constant during iterations over time, while term B_{ir} changes on consecutive iterations of the task. We propose the following learning rule for each i joint, as indicated in Expression (7):

$$\hat{B}_{i(r+1)} = \hat{B}_{ir} + P * e_{ir}, \quad (7)$$

where $P * e_{ir}$ is the convolution between the impulse response filter P and the error in iteration r. Among different filters, we chose the one given by Eq. (8):

$$P(s) = s^2 + (K_{vi} - \mu)s + (K_{pi} - \mu), \quad (8)$$

where μ is a constant.

P is a noncausal filter, and so it uses the errors of the previous iterations and its convergence depends on μ. Further specifics on the LF controller analysis are provided in Appendix A.

2.2 Unit learning machine

As a regression algorithm, the LWPR creates N linear local models y_k for achieving the function that best describes any set of training points (x_i, y_i). The LWPR uses the inputs $(x_1, x_2, \ldots, x_m)$ for each local model to produce the y_k signals and, by means of Eq. (9), to predict $\hat{y}$. In other words, the total output of the network is the weighted mean of all linear models.

$$\hat{y} = \frac{\sum_{k=1}^{N} p_k \overline{y}_k}{\sum_{k=1}^{N} p_k}. \tag{9}$$

In the LWPR, the first site of plasticity is the measure of the locality for each data point. This measure is obtained by means of a kernel function as in Eq. (10), a weighting kernel computes a weight $p(k, i)$ for each x_i data point according to the distance from the c_k center of the kernel in each k local unit. The weight is a measure of how often an item of x_i data falls into the region of validity for each linear model. The kernel function is defined as a Gaussian kernel:

$$p_k = exp\left(-\frac{1}{2}(x_i - c_k)^T D_k (x_i - c_k)\right), \tag{10}$$

where D_k is a positive definite matrix which is called *distance matrix*.

Regarding the learning process, the number of local models increases with the complexity of the input space. If a data sample falls into the validity region of a model, then its own distance matrix and regression parameters will be updated; furthermore, the update of each local model is independent from all the other models. As mentioned before, the inverse model is trained using a feedback error learning strategy. The LF controller converts trajectory errors into motor commands to be used as a training signal for the cerebellar network.

Comparing the LWPR processing unit and the cerebellar microcircuit shown in Fig. 1b, we take advantage of the LWPR kernels as granular and molecular layer microzones (complexes) in the cerebellum approach. According to Schweighofer et al. (2001), the function of the code in GC is that of providing sparse codes (that are informative of the MF inputs) to the subsequent neural layers to obtain a precise and stable cerebellar learning. So, this is what the LWPR provides by means of the kernels (10), it delivers a compact representation of the MF–GC synapses outputs in terms of RF filters. Then, the LWPR uses this kernel to produce the individual predictions y_k which as a whole represent the learning of the inverse model. Our C module is integrated with the LWPR to reproduce the function of the cerebellum. In this regard, the C module receives the preprocessed $p_k(t)$ signals as a bank of filters G_k (GC outputs), defined in (11), which are driven by the k_{th} PFs and the interneuron contributions. Then, the specific PF and the interneuron pathway to the Purkinje layer carry the $p_k(t)$ signal to the PC synapse:

$$p_k = G_k(x_1, x_2, \ldots, x_m), k = 1..N \tag{11}$$

PC output $z(t)$, defined in (12), is modeled as a weighted linear combination of the $p_k(t)$

$$z(t) = \sum_k w_k p_k(t). \tag{12}$$

The synaptic weights w_k of the kth PF-PC synapse (see Fig. 1b) are updated using the heterosynaptic covariance learning rule (13) (Sejnowski 1977) in the continuous form (Porrill and Dean 2007), and adjusted by an e_t teaching or error signal (the CF). For the adaptation of the synaptic weights, Fujita (1982) introduced the concept of adaptive filter in the framework of cerebellar modeling.

$$\delta w_k = -\beta e(t) p_k(t), \tag{13}$$

where β is a small positive learning rate and $e(t)$ is the error signal carried out by the CF. In this approach, $e(t)$ is the feedback error torque $\hat{u}_{fb_r}$.

In order to perform an optimal function approximation, the LWPR incrementally divides the input space into a set of RFs defined by the center c_k and a Gaussian area characterized by the particular kernel width D_k, as shown in Eq. (10). During each iteration, all RFs calculate their weight activation to assign the new input, x_i, to the closest RF and consequently, the center and the kernel width are incrementally updated. The optimized choice of centers and widths gives an optimal basis of RFs, so that the accuracy and the learning speed of the ULM are improved. In other words, Eq. (11) represents the bank of G_k filters for the GCs in the cerebellum and their response is both used to compute the cerebellar output $z(t) = u_c$, as defined in Eq. (12) and to update the synaptic weights (13).

From Fig. 3, we see that the internal model within the cerebellar cortex will retain the u_t global control torque as a target signal. Given that, Eq. (3) can now be completed:

$$u_t = M(Q_d)\ddot{Q}_d + V(Q_d, \dot{Q}_d) + G(Q_d) + \hat{u}_{fb} + \hat{u}_c. \tag{14}$$

The function defined in Eq. (15) represents the nonlinear function to be approximated by means of linear regressions, and it depends on the desired angular position, velocity, and acceleration, and on the real angular position and velocity of the joints of the arm.

$$u_l = \Phi(Q_d, \dot{Q}_d, \ddot{Q}_d, Q, \dot{Q}), \tag{15}$$

where function Φ can be learned online and offline. Further details about the method of learning are given in Subsect. 3.1.

3 Simulation results

We have verified the performance of the AFEL architecture in adapting to dynamic and kinematic changes of the controlled object on two physically realistic models of the LWR arm shown in Fig. 2. In the first setup, the LWR arm was simulated considering a reduced configuration to 3 DOFs to get fewer input dimensions to the machine learning engine. Specifically, the first (we will refer to it as Q_1), second (Q_2), and fifth joint (Q_3) have been used, while the others have been kept fixed. The three nonfixed joints used in our experiments are indicated in Fig. 2. This reduces the amount of training data required and expedites the initial learning process. Afterward, all the 7 DOFs of the LWR III shown in Fig. 2 were involved in the simulation, as described in Sect. 3.3. To simulate dynamic changes, we considered that the manipulated object was the last link of the arm, and so we changed the physical properties of the tip of the arm when emulating manipulation of different objects. Furthermore, to simulate a kinematic modification, we changed and fixed a certain orientation shift of the end-effector. Simulations were setup in the Matlab robotics toolbox (Corke 1996). The task for the experiments with the LWR arm was to follow a planned trajectory in a 3-dimensional task space.

3.1 Control performance evaluation

The robot end-effector traced out a target trajectory shown in Fig. 4b, defined by (16):

$$\begin{aligned} Q_1 &= Dsin(2\pi t), \\ Q_2 &= Dsin\left(2\pi t + \frac{\pi}{4}\right), \\ Q_3 &= Dsin\left(2\pi t + \frac{\pi}{2}\right), \end{aligned} \tag{16}$$

where D is a constant, and Q_1, Q_2, and Q_3 are the joint coordinates, respectively.

To approximate the nonlinear function described in Eq. (14), a sequence of 16 eight-like shaped movements was simulated to collect enough target points (8000) for training. Next, 15 iterations of the trajectory were repeated using the learned inverse dynamic model. An analytic model of the 3-DOF LWR arm (1) generated the feedforward joint data torques given the desired joint angles, velocities, and accelerations. In the training stage, the LWPR algorithm approximated Eq. (14) which contains the terms of the u_{fb} feedback joint torques and the u_c cerebellar joint torques besides the feedforward joint torques supplied by the analytic model. Then, the learned inverse dynamic model defined in Eq. (14) is tested: the analytic model is no longer used, the LWPR module predicted the joint torques to be applied to the robot plant, the cerebellum still optimized the execution of the trajectory and the LF supervised the system, and in the meanwhile learning of the u_t global torques was proceeding.The LWPR training took place for each DOF separately (a LWPR module for each i joint) with a training and a test set of [5 x number of joints] inputs (desired joint angles, velocities, and accelerations, $(Q_d, \dot{Q}_d, \ddot{Q}_d)$, respectively, and the current joint angles,and velocities, $(Q, and\, \dot{Q})$),respectively, and 1 target (joint torque u_{ti} of joint i) (according to Eq. (15)). As the LWPR has learned the inverse dynamics model, the movement is performed more precisely with a lower contribution command from the feedback and from the cerebellar circuitry. Therefore, LWPR works also as a memory consolidation module.

To evaluate the AFEL architecture performance, we examined how the tracking errors became compensated following the desired trajectory (16). In order to highlight the advantages of this novel adaptive control system, we set up six different architectures by substituting one block of the AFEL scheme for another, as shown in Fig. 4a. The thicker solid line in Fig. 4a is referred to the novel AFEL architecture performance (case 1). The LF controller was replaced with a high-gain PD (case 2) and with a low-gain PD (case 3) while the ULM module was substituted for an analytic dynamics method called Feed-Forward (FF) module (case 4). Lastly, both the LF controller and the adaptive ULM module were substituted for a high-gain PD (case 5) and for a low-gain PD (case 6). The used system accuracy measure is the normalized mean squared error (nMSE) between the desired joint angle in radians (Rad) and the actual joint angle in radians (Rad) obtained from the robot plant. The nMSE is defined as the MSE divided by the variance of the target data values. From Fig. 4a, we can see that the proposed AFEL architecture (case 1) achieves a very good performance with a low standard deviation. In order to guarantee a low tracking error in the system with a PD controller instead of the LF controller (cases 2 and 5), the PD gains had to be set to high values, which results in a potentially dangerous noncompliant movement because the manipulator could damage the environment if it comes into contact with it. As a result of this, the maximum torque (Nm) applied by joint actuators is too high.

Table 1 reflects this effect in terms of maximum torque u_t applied at each joint during a single trajectory. As a matter of fact, for ULM with high-gain PD architecture (case 2), the maximum torque gets up to 1000 Nm among the three joints, while for the AFEL architecture, the maximum torque was limited to around 200 Nm. To achieve the same rate of performance as the AFEL system (see Fig. 4), gains were multiplied by a factor of 250. Finally, by substituting the ULM module with an analytic model (case 4), the system still achieves good performance, but the standard deviation is higher, which means that the error is not equally diminished for all the joints at the same time. In other words, the cerebellum does optimize the miscalibration and gets adapted to novel dynamics.

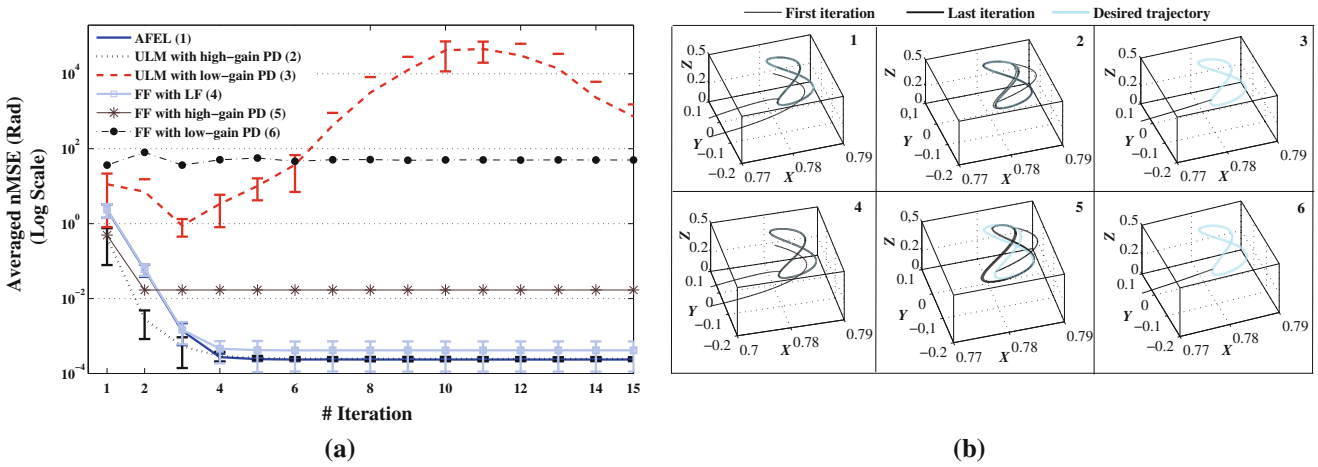


Fig. 4 Control architecture tracking performances manipulating a 6-kg load at the tip of the arm. Figure 4a displays normalized mean squared error (nMSE) averaged over three joints for six different architectures described in Subsect. 3.1. Error bars represent the standard deviation of the mean of the nMSE of the three joints. As a result of the simulation, Fig. 4b shows the eight-like-shaped figure (i.e., the desired and actual trajectories before and after learning, indicated as first iteration and last iteration relative to the left panel learning process) obtained after 15 trials for the six case studies (linked with the numbers on the top-right of the panels, referred to the control architectures indicated in Fig. 4a) in the task space. The low-gain PD controller (cases 3 and 6) yields a very large tracking error (therefore, the actual initial and final trajectories lay out of the plot)

Table 1 The first column contains the values of the maximum absolute torques applied at joints in adapting to the different dynamics models for the six different architectures

	Maximum absolute Torques (Nm)			RMS Nm)		
	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
AFEL (1)	88	206	106	52	112	56
ULM with high-gain PD (2)	620	1037	908	56	119	60
ULM with low-gain PD (3)	501	812	689	56	117	60
FF with LF (4)	116	213	130	52	112	56
FF with high-gain PD (5)	477	755	642	56	115	59
FF with low-gain PD (6)	74	177	59	42	92	31

Each case has been labeled with a number which is also used in Fig. 4. However, the second column contains the quadratic mean (RMS) of torques applied during all the iterations of the executed eight-like trajectory. Minimum absolute torques (Nm) (Q_1, Q_2, and Q_3) were always 0 in all the cases

During the experiment, we set the LF controller gains to very low values for a compliant control observing the sufficient condition provided by Nakanishi and Schaal (2004) to ensure stability of the FEL scheme into account.

Removing the cerebellar circuitry from the ULM module (LWPR alone), we obtain the result shown in Fig. 5, which is compared with the performance (dashed line) of the AFEL architecture. In both cases, the arm manipulates a 6-kg load at the grasper. This is the demonstration of how the cerebellum makes the LWPR learn an optimized inverse dynamics model of the robot arm and makes fine adjustments to the way the trajectory is performed. In fact, the nMSE is similar for each joint, as indicated by the error bars.

The second used performance measure is the mean absolute tracking error (MAE) in radians (Rad) defined in (17):

$$\left|Q_{d_t} - Q_t\right|, \tag{17}$$

which is averaged over a whole iteration. Q_{d_t} is the desired joint angle at time t, and Q_t is the actual joint angle at time t. From Fig. 6, we can see that the absolute tracking error decreases over the trials.

3.2 Dynamics and kinematics changes

The dynamics of the robot arm changes as the robot manipulates different objects or different contexts. In this section, four contexts are simulated by attaching objects with different masses at the tip of the arm. The masses are 2, 6, 8, and 10 kg, respectively. Fifteen iterations of the trajectory were executed using the inverse dynamics model of the four arm+object instantiation previously learned by the LWPR. We ran the experiment ten times with different initial positions around the trajectory, defined in Eq. (16), on each trial. We computed the robot arm tip position error in the

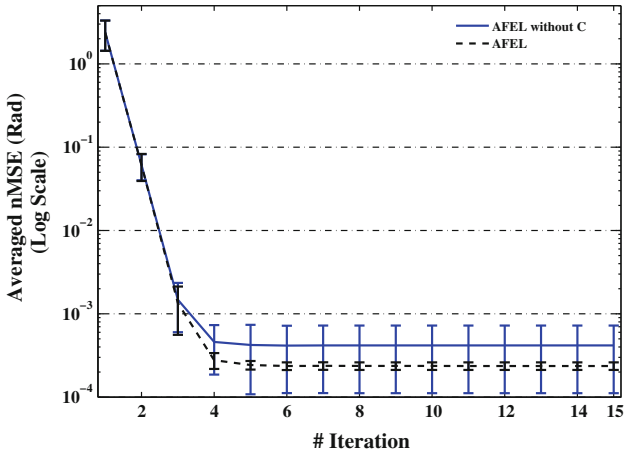


Fig. 5 The dashed line represents the normalized mean square error (nMSE) averaged over three joints related to the proposed AFEL architecture. The solid line shows the tracking error performance obtained by removing the cerebellar structure from the ULM module in the AFEL scheme. Comparing them, we make clear that the cerebellum drives the model abstraction engine (LWPR). In this way, the LWPR incrementally abstracts the whole model. Error bars represent the standard deviation above and below the mean of the nMSE of the three joints

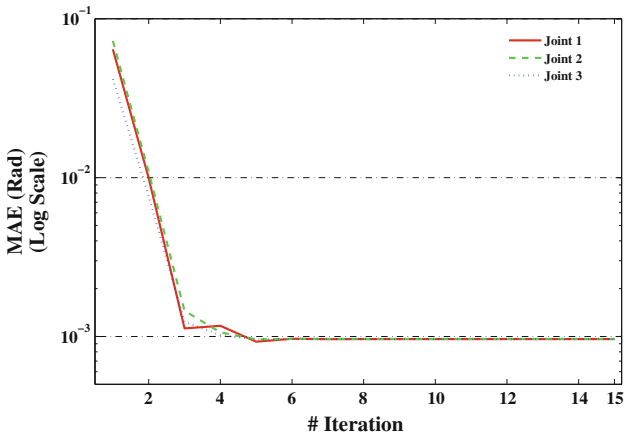
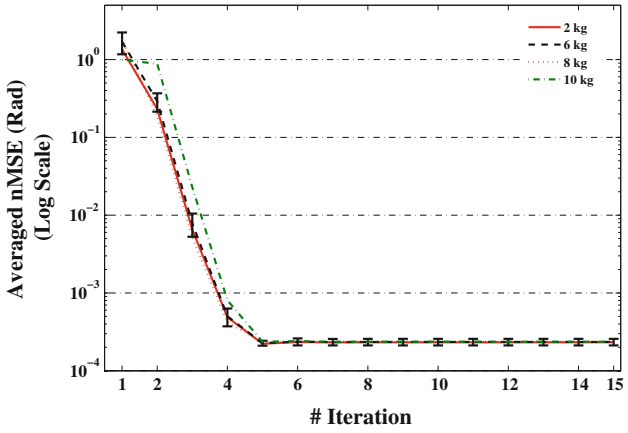
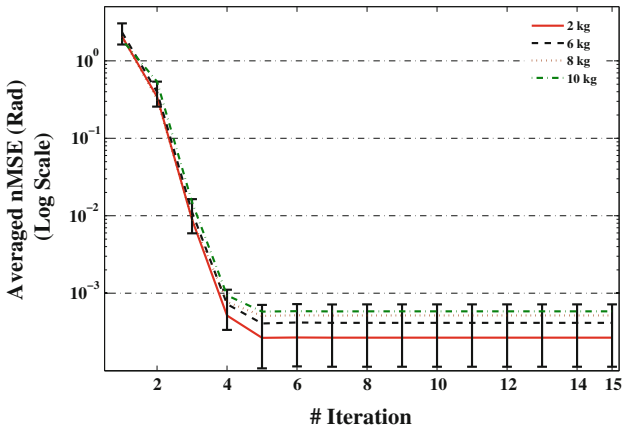


Fig. 6 AFEL architecture. Absolute tracking error, averaged over a whole iteration, manipulating a 6 kg load at the tip of the arm



(a) Outcome of the AFEL architecture.



(b) Outcome of the Feed-Forward architecture.

Fig. 7 Adaptation for the 3-DOF LWR arm using the AFEL architecture (a) and the Feed-Forward architecture (b) that contains an analytical model instead of the ULM module. Both figures show the average of the nMSE for three joints and over ten trials. Different traces indicate the response to different contexts. For the sake of clarity, error bars are plotted only for the 6 kg context and indicate the standard deviation between the trials.

different trials and averaged it over ten times. The gains of the LF controller have been set to the same values for the four objects.

Comparing Fig. 7a, related to the AFEL architecture, with Fig. 7b, related to the Feed-Forward architecture with the LF controller, the importance of the ULM module (LWPR + Cerebellum) becomes clear. In fact, in the second case, error bars are larger and the nMSE becomes higher as the load increases. However, results in Fig. 7a indicate the high quality of the estimate of the ULM output. In Table 2, we present the maximum torque applied at joints for adaptation to different contexts. As the load at the last joint is increased, compliance is gradually achieved by gradually increasing corrective joint torques. As mentioned before, we also tested the performance of the AFEL architecture in adapting to kinematics changes as well. The outcome is plotted in Fig. 8, which shows that performance is not affected either by changes in kinematics or by changes in dynamics. In this experiment, kinematics transformations applied at the robot plant consisted of different angles of fixed rotation of the end-effector ($\lambda = [30, 90]$).

3.3 Self-adaptive learning

Using an analytic method is not always possible to obtain a sufficiently accurate dynamics model which is needed for compliant robot control. In this case, it is necessary to adopt a new strategy. The u_{fb} feedback joint torques are given by the LF controller to control the arm. The LWPR modules receive their feedback command combined with its own prediction

Table 2 The first column contains the values of the maximum absolute torques applied at joints in adapting to the different dynamics models for the six different architectures

AFEL	Maximum absolute Torques (Nm)			RMS (Nm)		
	Q_1	Q_2	Q_3	Q_1	Q_2	Q_3
2 kg	80	189	76	45	97	35
6 kg	91	206	107	52	112	56
8 kg	99	231	122	56	128	65
10 kg	109	252	133	60	142	71

Each case has been labelled with a number which is also used in Fig. 4. However, the second column contains the quadratic mean (RMS) of torques applied during all the iterations of the executed eight-like trajectory. Minimum absolute torques (Nm) (Q_1, Q_2, and Q_3) were always 0 in all the cases

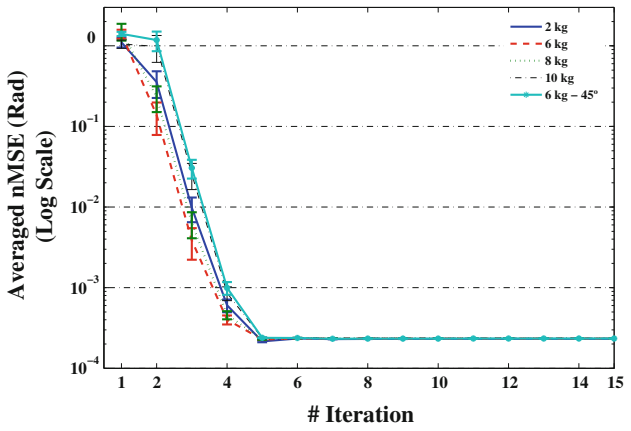
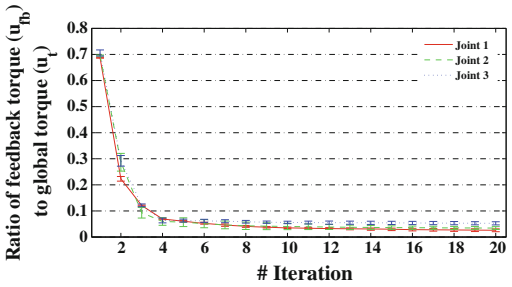


Fig. 8 Robustness of the AFEL architecture under kinematics and dynamics transformations. For the sake of clarity, only $\lambda = 45°$, which is representative of all values of λ tested, as kinematics transformation is plotted. The average nMSE for three joints is averaged over ten trials. Error bars indicate the standard deviation for ten trials

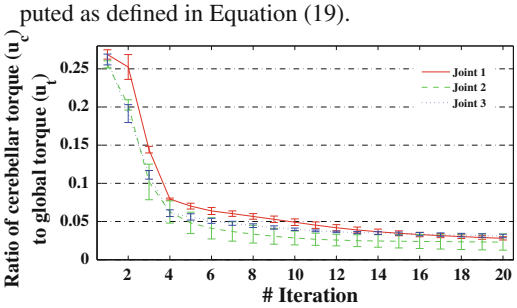
and the cerebellar output to form the feedforward motor command as a training signal. Then, in this experiment, there is no preliminary learning from an analytic model as in the previous approach. We repeat the trajectory 16 times for each context specified in Sect. 3.2. So, the LWPR still learns the u_{fb} global torque for the whole simulation while the LF adaptively controls the trajectory execution and the cerebellum optimizes the corrections. The task of this experiment is to follow the trajectory specified in Eq. (18)

$$\begin{aligned} Q_1 &= Asin(2\pi t), \\ Q_2 &= Asin\left(2\pi t + \frac{\pi}{4}\right), \\ Q_3 &= Acos\left(2\pi t + \frac{\pi}{2}\right). \end{aligned} \tag{18}$$

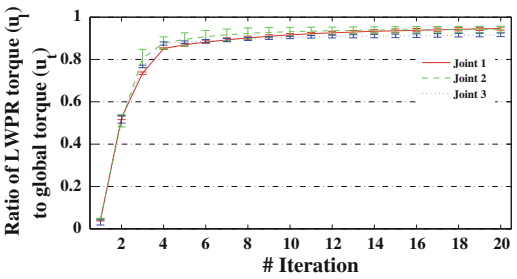
In order to evaluate the relative relevance of the feedback contribution along the learning process, the used performance measure is the ratio of torque components to the total joint torque applied to the robot plant. Therefore, we defined the ratios in Eqs. (19) and (20):



(a) Contribution of correcting feedback commands R_{fb} computed as defined in Equation (19).



(b) Contribution of correcting cerebellar commands R_c computed as defined in Equation (20).



(c) Indirect measurement of how well the inverse dynamic model learned by the LWPR approximates the actual dynamics. Values are computed as defined in Equation (21).

Fig. 9 Ratios of individual joint torque contributions to the global torque. Results are averaged over four trials (four contexts), and the error bars indicate the standard deviation between the trials

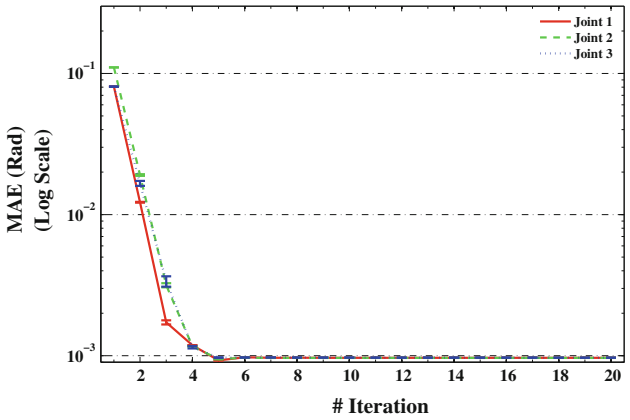


Fig. 10 Mean Absolute Error (MAE) averaged over four trials (four contexts indicated in 2)

$$R_{fb} = \frac{u_{fb}}{u_{fb} + u_{ff}}, \tag{19}$$

$$R_c = \frac{u_c}{u_{fb} + u_{ff}}. \tag{20}$$

Equation (19) represents the ratio of u_{fb} feedback torque to the u_t global torque, and Eq. (20) is the ratio of u_c cerebellar torque to the u_t global torque.

Figures 9a and 9b show how the average values (within each iteration) of these ratios evolve along the learning process. If the learned inverse model is accurate, then the ratios will be small as the error-correcting torque decreases over consecutive iterations. At the beginning of the simulation, the amount of ratio R_{fb} is higher than R_c, which means that the LF controller output contributes more to the global torque than the cerebellar torque and decreases significantly according to the reduction of errors. However, the second ratio, R_c, depends on the LWPR learning performance because they are connected by the RF weights of the LWPR local models which are the cerebellar granular weighting kernels. As a matter of fact, R_c decreases (see Fig. 9b) as the LWPR incorporates u_{fb} and u_c to its global output torque during the learning process.

Figure 9c shows the ratio of the LWPR torque u_l to the global torque u_t, as defined in Eq. (21):

$$R_l = \frac{u_l}{u_{fb} + u_{ff}}. \tag{21}$$

We can see that the LWPR algorithm progressively learns the u_t global torque and therefore it really acts as internal model for the inverse dynamics of the robot arm.

In short, the more accurate the learned model is, the finer will be the contribution of the cerebellum, because the LWPR optimally allocates the RFs for an efficient input mapping. In any case, both torque correction quantities, u_{fb} and u_c, vary depending on the nature of the system and on the error

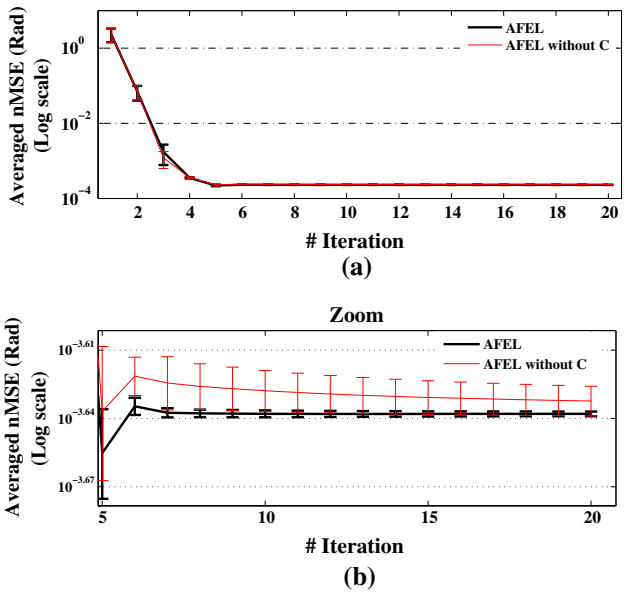


Fig. 11 Averaged nMSE over three joints. Error bars represent the standard deviation above and below the mean of the nMSE of the three joints. The thicker solid line is related to the proposed AFEL architecture, while the other is related to the AFEL architecture without the cerebellar structure in the ULM module. Comparing them, we notice that the cerebellum optimizes the tracking error performance and drives the joints to a better and faster convergence

because of miscalibrations, contexts, noise, etc. In all cases, the u_{fb} feedback component and the u_c cerebellar torque will decrease as the LWPR incorporates their contributions in its internal model as a memory consolidation process. Furthermore, as displayed in Fig. 10, the tracking absolute error is very low and with low variance between trials (contexts). Again, the fact that the error slopes down is a result of the control stability provided by the LF controller and of the cerebellar optimization that improved the dynamic inverse model to be learned. Comparing Figs. 6 and 10, we find that the performances are similar, which means that the self-adaptive learning works, and it is of high interest in case of unavailability of an analytic dynamics model. In addition, we can say that the cerebellum not only drives the LWPR learning engine to acquire an optimized internal model, but also contributes to deliver finer and more effective corrections for all contexts. For this purpose, we compare the performances of the proposed AFEL system with an identical one in which the ULM module does not contain the cerebellar microcircuit. Results are plotted in Fig. 11.

Finally, we have verified that the self-adaptive learning works efficiently on a more complex robotic platform too. For this purpose, we have repeated the same experiment for the 7-DOF LWR arm, measuring the outcome performance in terms of nMSE and computing the ratios described in Eqs. (19), (20), and (21). The eight-like-shaped target trajectory to be followed by the arm tip is defined in Eq. (22):

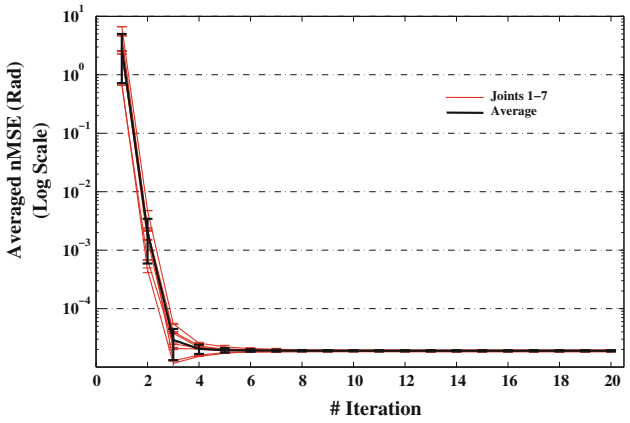


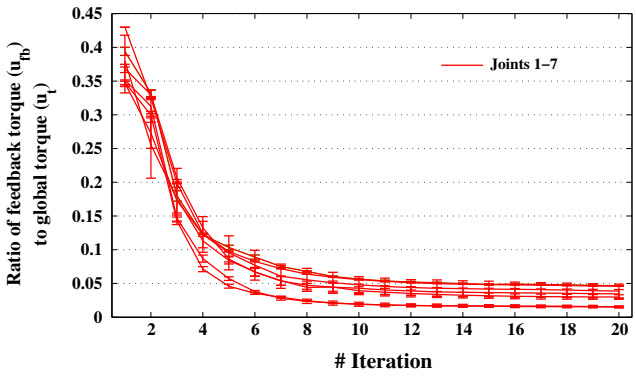
Fig. 12 nMSE averaged over four trials (four contexts indicated in 2). The thicker, darker line is the average over the 7 joints

$$y = 0.15sin(2t),$$
$$z = 0.6 + 0.2cos(t), \tag{22}$$

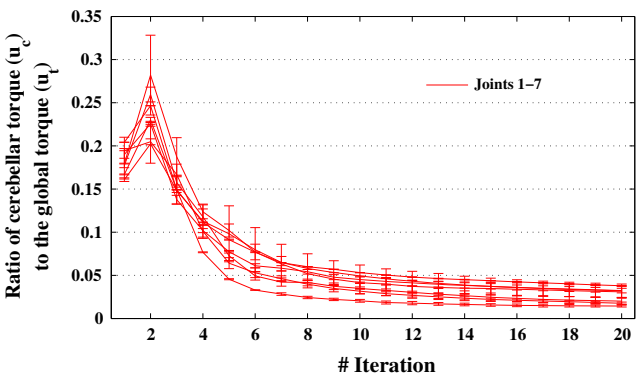
and the variable x is a constant.

Results in Fig. 12 indicate that the AFEL architecture also works for high DOFs. The nMSE is low and after five iterations, the system's behavior becomes stable. The LWPR has approximated the dynamics model of the LWR arm well achieving high accuracy. Figures 13a and 13b shows the ratios of each individual component torque, u_{fb} and u_c, with respect to the global joint torque, u_t, while Fig. 13c reveals that the LWPR output increases according to its gradual learning of the global torque, u_t.

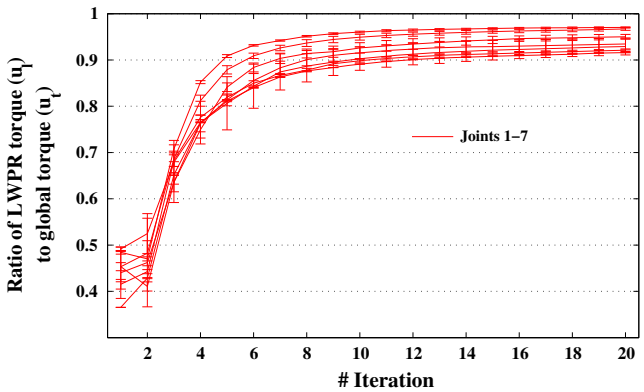
The behavior of the AFEL scheme has been studied in a noisy scenario by using uniform and Gaussian additive noise in the inputs of the system (at the inputs of the LWPR learning engine) as represented in Figures 14 and 15 related to the 3DOFs and 7DOFs configurations, respectively. Noise causes a deviation in the actual trajectory. In the framework of a biological system, it seems that the cerebellar circuitry helps to reduce the noise introduced in the MFs (which are highly stochastic) facilitating learning in the molecular and Purkinje layer (Philipona and Coenen 2004). Both Figures 14 and 15 show the nMSE evolution obtained when the robot



(a) Contribution of correcting feedback commands R_{fb} computed as defined in Equation (19).



(b) Contribution of correcting cerebellar commands R_c computed as defined in Equation (20).



(c) Contribution of LWPR feed-forward commands R_l to the u_t global torque computed as defined in Equation (21).

Fig. 13 Ratios of individual torque contributions to the u_t global torque. Results are averaged over the four trials (four contexts), and the error bars indicate the standard deviation between the trials

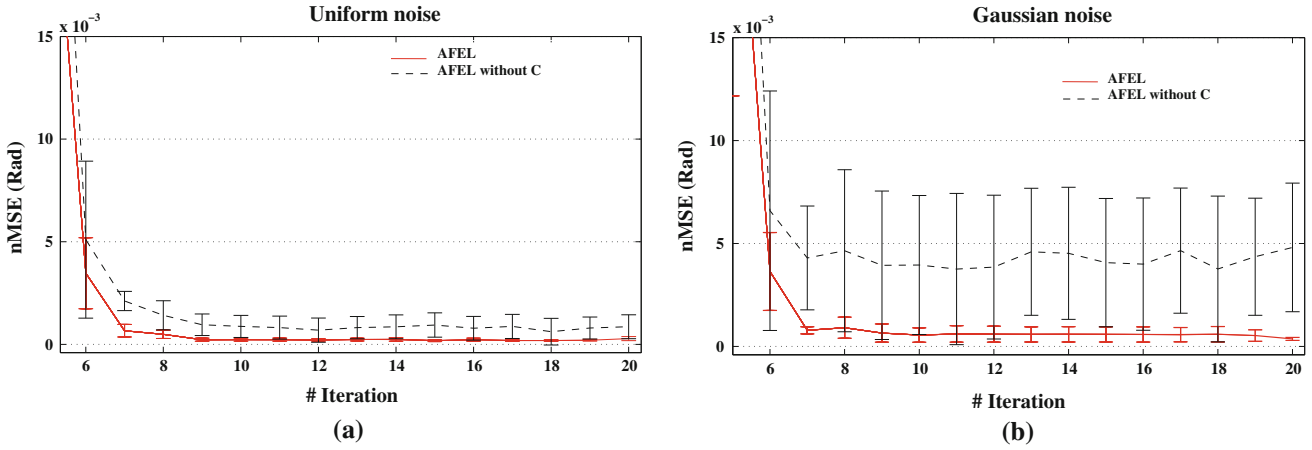


Fig. 14 Accuracy evolution of the AFEL architecture on a 3-DOF configuration when introducing a uniform noise **(a)** and a Gaussian noise **(b)** on input signals to the system. Graphs represent the averaged nMSE in radians over three joints of the robot loaded with a 10-kg weight during the learning process of the eight-like-shaped trajectory execution. The zoom in the nMSE axis highlights the difference between curves

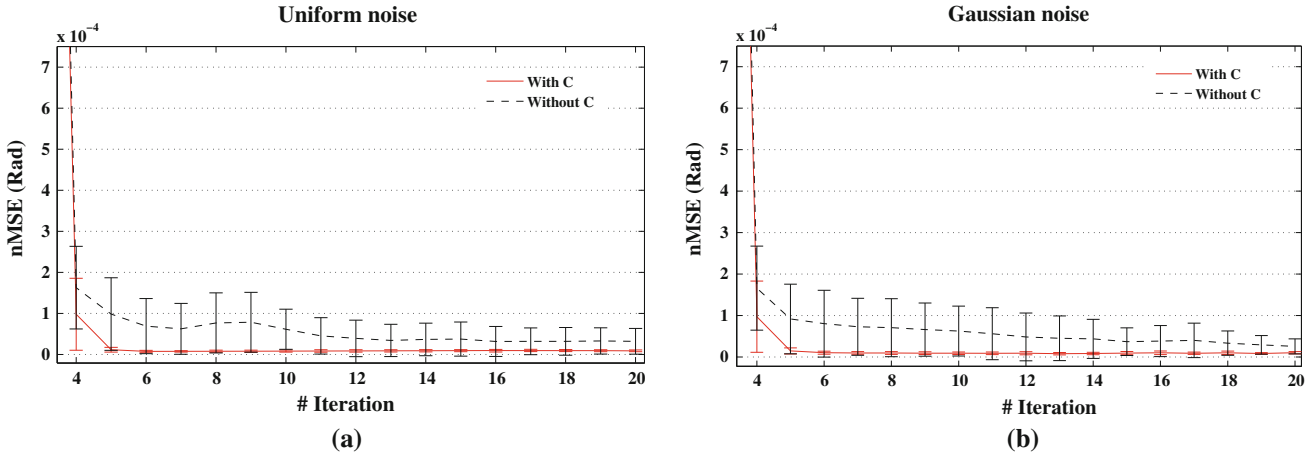


Fig. 15 Accuracy evolution of the AFEL architecture on a 7DOFs configuration when introducing a uniform noise **(a)** and a Gaussian noise **(b)** on input signals to the system. Graphs represent the averaged nMSE in radians over seven joints of the robot loaded with a 10-kg weight during the learning process of the eight-like-shaped trajectory execution. The zoom in the nMSE axis highlights the difference between curves

end effector was loaded with 10 kg to increase the inertia, and the added noise was set to 16 dB using a uniform (a) and Gaussian distribution (b), respectively. As a result of this, the AFEL architecture remains stable against this input noise when the C module is embedded in it, while the LWPR alone does not accomplish the joint convergence and the stability. In fact, the cerebellar output allows a good precision, and joint error remains delimited around mean values (i.e., smaller standard deviation).

More importantly, the results obtained in this study indicate that the AFEL scheme (including a cerebellar module in the ULM) is scalable in terms of number of joints. For the 7-DOF LWR arm, we obtained good results in terms of error, just as for the 3-DOF case. However, it should be noted that the dynamics of a real system will significantly be more complicated than the simulated dynamics, as there are important nonlinear effects that are not simulated, such as actuator dynamics or elasticity. Furthermore, as already indicated by other authors, it has been shown that learning of dynamics using LWPR on real-world high DOF robotic platforms works very efficiently (Vijayakumar et al. 2005).

4 Conclusions

We implemented a model for the motor control of robotic arm movements in which machine learning and biologically inspired approaches co-exist and complement each other. The presented Adaptive Feedback Error Learning scheme (AFEL), which takes advantage of the connection between the accurate regression method LWPR and a basic cerebellar structure, works properly. Furthermore, the cerebellar

module takes full advantage of the LWPR for efficiently abstracting the high dimensional input space.

A potential role of the granular and molecular layers in biologically plausible cerebellar models is to provide accurate signals to the PCs for improving the learning of the current model. These signals seem to influence the DCN synapses capability to consolidate the learning (Wulff et al. 2009; Attwell et al. 2002; Boyden et al. 2004). In our model, the input LWPR RFs are used as a representation of the granular and molecular layers delivering clean and accurate signals to the Purkinje layer. Therefore, LWPR provides optimal input representation to the Purkinje layer in terms of neural resources (it adapts its neural resources incrementally and according to the input data structure). The importance of this efficient and clean contribution has been evaluated recently (Wulff et al. 2009; Honda et al. 2010) and other authors, such as (Schweighofer et al. 2001) and (Porrill and Dean 2007) hypothesized that the cerebellar learning is facilitated by optimizing the choice of the centers and the basis of RFs at the granular layer. As a matter of fact, LWPR creates around 60 locally linear models for the 7-DOF robot arm, and it allows for the selection of only predictions from those who elicit great activation for a query point.

It is also important to note that the LWPR also works as a memory consolidation module. Therefore, the cerebellar module with its learning rule is focused on short-term adaptation, while long-term memory consolidation takes place at the LWPR module. Thus, in terms of short/long-term learnings by analogy with biological systems, our cerebellar module, receiving inputs from the LWPR RFs, represents the MF-GrC/interneurons-PC pathway for short-term learning, while the LWPR adaptation kernels represent the MF-DCN adaptive pathway which is responsible of long-term learning (memory consolidation) (Wulff et al. 2009; Masuda and Amari 2008).

We exploited the LWPR characteristics to acquire the dynamics of a robot arm through the learning process as the movement is repeated. Once the inverse model is learned, the system can perform the task precisely, and few correction forces are required, thus increasing the compliance. In order to achieve compliant movements and ensure the system stability, high feedback gains must be avoided, specially in biological systems (or robotic platforms in human machine interaction tasks). To avoid this, we propose the LF controller, which adapts the error-correcting feedback over consecutive iterations of the same task. The LF controller supplies the error to the cerebellar network. Furthermore, the LF controller will accurately guide the LWPR during the learning process using very low gains. Results show that the global architecture has a compliant performance which is suitable for robotic systems in human environments. Haddadin et al. (2007) evaluated the influence of the mass and velocity of DLR-LWRIII in resulting injuries on human bodies. Their impact tests were carried out using the Head Injury Criterion (HIC) as it is the most prominent indicator of head injury in automobile crash-testing; the results of this test suggest that a robot, even with arbitrary mass moving not much faster than 2 m/s is not able to become dangerous to a nonclamped human head with respect to typical severity indices. In addition to this, their investigation revealed that the inertia properties of the LWRIII allow an impact velocity of up to 1 m/s without leading to soft-tissue injuries. With regard to our system, because the linear velocity is the cross-product between the angular velocity and the position of the end-effector with respect to origin, we have considered the worst case, i.e., the longest link and the maximum angular velocity, to simplify the computation. The maximum linear velocity obtained during the self-adaptive experiment for the 3-DOF LWR arm manipulating a 10-kg load at the last joint is 0.62 m/s, and therefore the system performs in a compliant manner.

The performance obtained by the AFEL scheme in terms of error after learning is remarkable compared to other approaches (Nguyen-Tuong and Peters 2008; Lonini et al. 2009). This is of high interest taking also into account that in this architecture, the LWPR does not require an analytic preliminary dynamic model to learn from it, as it learns directly from the feedback torques and the cerebellar compensatory torques. As a matter of fact, LWPR works as an internal model abstraction kernel whose learning process is guided by the LF instead of having a reference analytic model.

Porrill and Dean (2007) mentioned the motor error problem due to the complexity on the reference structures used to compute the error for the forward connectivity. This problem particularly affects biological nonlinear motor systems as the number of the reference structures is multiplicative in the dimension of the control and sensor space. In our feedback error learning approach, we addressed the motor error problem using the LF controller, we proved its performance in the task of a simulated 3 and 7-DOF robot arm. We showed that the combination of feedback and feedforward estimates does offer considerable advantages for robust online control. Furthermore, we ensured accuracy and enhanced the speed of learning by optimizing the choice of the centers and transforming to an optimal basis of RFs through the LWPR algorithm.

Schweighofer et al. (2001) proposed a diagram of the model of cerebellar control two-joint arm movements in which the cerebellum learns how to compensate for interaction torques that occur during reaching movements. In analogy with their approach, in our control system, there are three motor commands: feedforward motor commands (by the forward internal model), feedback torques (by the LF controller), and the cerebellar compensatory torques (by the C module) which are summed and sent to the robot arm plant. The cerebellar torque values are necessary for precise control and the cerebellar network is embedded in the control model.

In fact, the comparison between two cases, including and not including the C module in the system, shown in Figures 14 and 15, respectively, makes clear the important role of the cerebellar C module achieving high robustness against noise. Owing to the large number of DOFs and the pervasive nonlinearities of the seven degrees of freedom human arm, an internal model of the arm's dynamics is an extremely complex mapping between kinematic and dynamic variables, and thus requires a large number of encoding states (Schweighofer et al. 2001). Despite this, we demonstrated that we achieved very good performances with a small number of states (or GCs) for a 7-DOF robot arm.

In conclusion, to the best of our knowledge, the presented AFEL model provides significant advantages for adaptive motor control. AFEL uses constrained torques, which makes the approach appropriate for compliant movements. It provides highly accurate movement capabilities (even in the presence of disturbances of the initial dynamics and kinematics of the "robot+object" plant), i.e., low errors for all joints, even similar to other approaches using control strategies based on high gains. It achieves even better results in plants of high DOFs, and all the joints converge faster to the minimum error. Finally, AFEL uses an adaptive learning module to feed the LWPR component, making this approach useful even for robotic plants for which the analytic dynamics or kinematics are only roughly known. This hybrid scheme based on a machine learning engine (LWPR) and a bio-inspired component (cerebellar module) efficiently uses the LWPR component to optimize the input space representation and also efficiently uses the cerebellar-like structure to integrate different input-driven contributions for obtaining accurate corrective commands.

Acknowledgments This study has been supported by the EU project SENSOPAC (IST-028056) and by the Sardinian Master and Back Program (P.O.R. FSE 2007-2013).

Appendix: LF controller details

Substituting the Laplace transform of Eq. (6) in the Laplace transform of Eq. (7), we obtain Eq. (23):

$$\hat{B}_{i(k+1)}(s) = \hat{B}_{ik}(s) + P(s)H(s)[B_i(s) - \hat{B}_{ik}(s)], \quad (23)$$

where

$$H(s) = \frac{1}{s^2 + K_{vi}(s) + K_{pi}}. \quad (24)$$

Substituting the following Eq. (25) in Eq. (23)

$$G(s) = 1 - P(s)H(s), \quad (25)$$

we obtain Eq. (26)

$$\hat{B}_{i(k+1)}(s) = G(s)\hat{B}_{ik}(s) + \hat{B}(s)[1 - G(s)]. \quad (26)$$

Starting with D_{i0}, after k iterations we will get Eq. (27):

$$\hat{B}_{ik}(s) = B_i(s) + AG^k(s), \quad (27)$$

where A is a constant. The convergence of the learning algorithm depends on the factor $G^k(s)$ in Eq. (27); The convergence will occur if $G^k(s)$ approaches zero. In this case, $\hat{B}_{ik}(s) \rightarrow B_i(s)$, and Expression (5) will be true with its right expression equal to zero. The inverse Laplace transform of $G^k(s)$ is defined by Eq. (28)

$$g_k(t) = L^{-1}[G^k(s)]. \quad (28)$$

If we choose an appropriate filter $P(s)$, (8), then we would obtain

$$lim_{k\rightarrow\infty} |h_k(t)| = 0, \quad (29)$$

with which the convergence is guaranteed.

References

Albus JS (1971) A theory of cerebellar function. Math Biosci 10(1–2): 25–61

Albus JS (1975) A new approach to manipulator control: the cerebellar model articulation controller (CMAC). J Dyn Syst Meas Control 97(3):220–227

Atkeson CG, Hale JG, Pollick F, Riley M, Kotosaka S, Schaul S, Shibata T, Tevatia G, Ude A, Vijayakumar S, Kawato E, Kawato M (2000) Using humanoid robots to study human behavior. IEEE Intell Syst Appl 15(4):46–56

Attwell P, Cooke S, Yeo C (2002) Cerebellar function in consolidation of a motor memory. Neuron 34(6):1011–1020

Boyden E, Katoh A, Raymond J (2004) Cerebellum-dependent learning: the role of multiple plasticity mechanisms. Neuroscience 27(1):581–609

Carrillo R, Ros E, Boucheny C, Coenen O (2008) A real-time spiking cerebellum model for learning robot control. Biosystems 94(1–2):18–27

Corke PI (1996) A robotics toolbox for matlab. IEEE Robotics Autom Mag 3(1):24–32

Craig JJ (2005) Introduction to robotics: mechanics and control, 3rd edn. Pearson/Prentice Hall, Upper Saddle River

Dean P, Porrill J, Ekerot C, Jörntell H (2010) The cerebellar microcircuit as an adaptive filter: experimental and computational evidence. Nat Rev Neurosci 11(1):30–43

Fujita M (1982) Adaptive filter model of the cerebellum. Biol Cybern 206(3):195–206

German Aerospace Center (2011) DLR Light-Weight Robot (LWR). http://www.dlr.de/rm/en/desktopdefault.aspx/tabid-3803/6175_read-8963/. Accessed 11 August 2011

Gomi H, Kawato M (1992) Adaptive feedback control models of the vestibulocerebellum and spinocerebellum. Biol Cybern 68(2):105–114

Haddadin S, Albu-SchSffer A, Hirzinger G (2007) Safe physical human-robot interaction: Measurements, analysis and new insights. In: Kaneko M, Nakamura Y (eds) ISRR, Springer, Springer Tracts in Advanced Robotics, vol 66, pp 395–407

Haith A, Vijayakumar S (2009) Implications of different classes of sensorimotor disturbance for cerebellar-based motor learning models. Biol Cybern 100(1):81–95

Hirzinger G, Butterfaß J, Fischer M, Grebenstein M, Hähnle M, Liu H, Schäfer I, Sporer N (2000) A mechatronics approach to the design of light-weight arms and multifingered hands. In: ICRA, pp 46–54

Honda T, Yamazaki T, Tanaka S, Nishino T (2010) A possible mechanism for controlling timing representation in the cerebellar cortex. In: International Symposium on Neural Networks, pp 67–76

Ito M (2000) Mechanisms of motor learning in the cerebellum. Brain Res 886(1-2):237–245

Ito M (2008) Control of mental activities by internal models in the cerebellum. Nat Rev Neurosci 9(4):304–313

Jordan MI (1996) Computational aspects of motor control and motor learning. In: Heuer, H and Keele, S (ed) Handbook of perception and action: Motor Skills, Academic Press, New York, vol 2, pp 71–120

Jordan MI, Rumelhart DE (1992) Forward models: supervised learning with a distal teacher. Cognit Sci 16:307–354

Kawato M (1990) Feedback-error-learning neural network for supervised motor learning. In: R Eckmiller (ed) Advanced neural computers, Elsevier, North-Holland, pp 365–372

Kawato M (1999) Internal models for motor control and trajectory planning. Curr Opin Neurobiol 9(6):718–727

Lonini L, Dipietro L, Zollo L, Guglielmelli E, Krebs HI (2009) An internal model for acquisition and retention of motor learning during arm reaching. Neural Comput 21(7):2009–2027

Luque N, Garrido J, Carrillo R, Coenen O, Ros E (2011a) Cerebellarlike corrective model inference engine for manipulation tasks. IEEE Trans Syst Man Cybern B 41(5):1299–1312

Luque N, Garrido J, Carrillo RR, Tolu S, Ros E (2011b) Adaptive cerebellar spiking model embedded in the control loop: context switching and robustness against noise. Int J Neural Syst 21(5):385–401

Luque NR, Garrido JA, Carrillo RR, Coenen OJMD, Ros E (2011c) Cerebellar input configuration toward object model abstraction in manipulation tasks. IEEE Trans Neural Netw 22(8):1321–1328

Marr D (1969) A theory of cerebellar cortex. J Physiol 202:437–470

Masuda N, Amari S (2008) A computational study of synaptic mechanisms of partial memory transfer in cerebellar vestibulo-ocular-reflex learning. J Comput Neurosci 24(2):137–156

Miyamura A, Kimura H (2002) Stability of feedback error learning scheme. Syst Control Lett 45(4):303–316

Nakanishi J, Schaal S (2004) Feedback error learning and nonlinear adaptive control. Neural Netw 17(10):1453–1465

Nguyen-Tuong D, Peters J (2008) Learning robot dynamics for computed torque control using local gaussian processes regression. In: Proceedings of the 2008 ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems, IEEE Computer Society, Washington, DC, USA, pp 59–64

Philipona D, Coenen OJMD (2004) Model of granular layer encoding of the cerebellum. Neurocomputing 58(60):575–580

Porrill J, Dean P (2007) Recurrent cerebellar loops simplify adaptive control of redundant and nonlinear motor systems. Neural Comput 19(1):170–193

Porrill J, Dean P, Stone J (2004) Recurrent cerebellar architecture solves the motor-error problem. Proc Biol Sci 271(1541):789–796

Schaal S, Atkeson CG, Vijayakumar S (2002) Scalable techniques from nonparametric statistics for real time robot learning. Appl Intell 17(1):49–60

Schweighofer N, Spoelstra J, Arbib MA, Kawato M (1998) Role of the cerebellum in reaching movements in humans. II. A neural model of the intermediate cerebellum. Eur J Neurosci 10(1):95–105

Schweighofer N, Doya K, Lay F (2001) Unsupervised learning of granule cell sparse codes enhances cerebellar adaptive control. Neuroscience 103:35–50

Sejnowski TJ (1977) Storing covariance with nonlinearly interacting neurons. J Math Biol 4:303–321

Shibata T, Schaal S (2001) Biomimetic gaze stabilization based on feedback-error-learning with nonparametric regression networks. Neural Netw 14(2):201–216

Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199–222

Van der Smagt P (1998) Cerebellar control of robot arms. Connect Sci 10:301–320

Van der Smagt P, Groen F, Schulten K (1996) Analysis and control of a rubbertuator arm. Biol Cybern 75(5):433–440

Vijayakumar S, Schaal S (2000) Locally weighted projection regression: Incremental real time learning in high dimensional space. In: ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 1079–1086

Vijayakumar S, D'Souza A, Schaal S (2005) Incremental online learning in high dimensions. Neural Comput 17(12):2602–2634

Williams CKI, Rasmussen CE (1996) Gaussian processes for regression. In: Advances in neural information processing systems 8. MIT press, pp 514–520

Wolpert DM (1997) Computational approaches to motor control. Trends Cogn Sci 1(6):209–216

Wolpert DM, Miall RC, Kawato M (1998) Internal models in the cerebellum. Trends Cogn Sci 2(9):338–347

Wulff P, Schonewille M, Renzi M, Viltono L, Sassoè-Pognetto M, Badura A, Gao Z, Hoebeek FE, Dorp Svan , Wisden W, Farrant M, De Zeeuw CI (2009) Synaptic inhibition of purkinje cells mediates consolidation of vestibulo-cerebellar motor learning. Nat Neurosci 12(8):1042–1049

Yamazaki T, Tanaka S (2007) The cerebellum as a liquid state machine. Neural Netw 20(3):290–297