International Journal of Neural Systems, Vol. 23, No. 3 (2013) 1350010 (15 pages)

DOI: 10.1142/S012906571350010X

World Scientific
www.worldscientific.com

ADAPTIVE AND PREDICTIVE CONTROL OF A SIMULATED ROBOT ARM

SILVIA TOLU
CITIC-Department of Computer Architecture and Technology
University of Granada, Periodista Daniel Saucedo s/n
18014 Granada, Spain
stolu@atc.ugr.es

MAURICIO VANEGAS
PSPC-Group, Department of Informatics, Bioengineering
Robotics and Systems Engineering (DIBRIS)
University of Genoa, Via Opera Pia 13, 16145 Genova, Italy
mauricio.vanegas@unige.it

JESÚS A. GARRIDO
Consorzio Interuniversitario per le Scienze Fisiche della Materia (CNISM)
Via Bassi 6, I-27100 Pavia, Italy
jesus.garrido@unipv.it

NICETO R. LUQUE* and EDUARDO ROS†
CITIC-Department of Computer Architecture and Technology
University of Granada, Periodista Daniel Saucedo s/n
18014 Granada, Spain
**nluque@atc.ugr.es*
†*eros@atc.ugr.es*

Accepted 19 February 2013
Published Online 25 March 2013

In this work, a basic cerebellar neural layer and a machine learning engine are embedded in a recurrent loop which avoids dealing with the motor error or *distal error* problem. The presented approach learns the motor control based on available sensor error estimates (position, velocity, and acceleration) without explicitly knowing the motor errors. The paper focuses on how to decompose the input into different components in order to facilitate the learning process using an automatic incremental learning model (locally weighted projection regression (LWPR) algorithm). LWPR incrementally learns the forward model of the robot arm and provides the cerebellar module with optimal pre-processed signals. We present a recurrent adaptive control architecture in which an adaptive feedback (AF) controller guarantees a precise, compliant, and stable control during the manipulation of objects. Therefore, this approach efficiently integrates a bio-inspired module (cerebellar circuitry) with a machine learning component (LWPR). The cerebellar-LWPR synergy makes the robot adaptable to changing conditions. We evaluate how this scheme scales for robot-arms of a high number of degrees of freedom (DOFs) using a simulated model of a robot arm of the new generation of light weight robots (LWRs).

Keywords: Light weight robot; recurrent control architecture; locally weighted projection regression; adaptive learning.

1. Introduction

In this paper, we integrate different elements into an approach for adaptive and predictive control in manipulation tasks. In biological systems, the cerebellum seems to play a key role in performing accurate and coordinated movements. It is involved in the control of actions and in the acquisition of specific motor skills.[1–3] The behavior of the cerebellum has been commonly emulated in a feed-forward control architecture by artificial neural networks (ANNs)[4] based on feedback error learning (FEL), where it delivers feed-forward corrective terms to the crude commands from the motor cortex[5–8] and the teaching signal is computed by the difference between actual and correct/desired motor commands, that is the motor error. Nevertheless, the correct motor command is typically unknown; only sensory errors are available, and the way to use this information for motor learning represents the so-called *distal error* problem.[9] Porrill and Dean stated in Ref. 10 that this kind of approach requires complex neural structures to estimate the motor error; thus they advocated using the *recurrent control architecture*. In fact, unlike in the feed-forward approach, the recurrent control architecture uses sensory-based teaching signals for adaptively adjusting the cerebellar weights.[11] These signals are physically available signals[10,12] and represent sensory mismatch between the desired and actual movement. Therefore, this configuration avoids the *distal error* problem.

Evidence from neurophysiology and computational studies support that, in the cerebellar cortex, two different types of internal models are allocated.[13–15] They are the inverse and forward models which are adjusted over time with supervised learning to facilitate precise coordinated movements[12] and to increase the system's control compliance.[16] The inverse internal model reproduces the inverse dynamics model of a body part,[15] i.e. for a desired motion described in joint-coordinates, it produces the motor command in terms of torques to be applied. The second one (forward model) reproduces the forward dynamics; in this way, for example, it predicts the next state of the arms (given both the current state and the applied torque values) and plays an important role in arm control.[9,17] Miall and Wolpert suggested in Ref. 18 that forward models are employed by the sensory-motor system to predict the consequences of movements based on the efference copy of the motor command and the current state of the body.

Porrill and Dean in Ref. 10 modeled the cerebellum as a set of complex adaptive filters in which the decomposition of the input into different components takes place at the granular layer. They argued that the choice of an appropriate basis (for these adaptive filters) significantly affects both the learning speed and the accuracy. In our approach, we adopt a recurrent adaptive control architecture, in which the cerebellar module delivers corrective terms in the sensory space. We simulate the cerebellar module as a single neural layer (abstracting the Purkinje cell (PC) layer) that adapts its weights using the covariance learning rule.[19] This single neural layer performs a weighted sum of inputs from adaptive filters which abstract a rich representation of the input sensory-motor space. Instead of building a granular layer for these adaptive filters, we make use of the local weight projection regression (LWPR) algorithm[20,21] that internally builds up a set of filters to acquire a compact and optimized representation of the input. Furthermore, the LWPR module plays the role of a forward model that delivers sensory estimates (predictive sensory consequences of the motor commands).

LWPR has already been used for online incremental learning in robotic platforms,[21–24] since it exploits spatially localized linear models to approximate nonlinear functions at a low computational cost. The evaluation of the prediction value is rather fast, allowing real-time learning. Furthermore, the incremental training of the LWPR allows the acquisition and retention of new models along the lifelong use of the robot. The major strength of the LWPR is that it can cope with highly redundant and irrelevant data input dimensions, without any prior knowledge of the data. This is so because it uses an incremental version of the partial least squares regression (PLS). On the other hand, Gaussian process regression (GPR)[25] or support vector regression (SVR)[26,27] have the advantage of not depending on many parameters and they are, therefore, easier to tune.[28] A revision of different regression methods for robotics can be found in Ref. 28. LWPR has been convincingly applied to model kinematics and dynamics (forward and inverse) of large mechanical systems.[28] In particular, considering the forward

dynamic model learning case, LWPR has been used to learn the forward dynamic model of a two DOFs plant robot.[29]

With regard to the control, we use a simulated model of a robot arm of the new generation of light weight robots (LWRs)[30,31] whose major drawback is that its complex dynamics are difficult to control with traditional methods, since the movement is affected by the state variables of all the joints and control becomes very complex and highly nonlinear.[32] Using a simple linear control, these nonlinearities can be compensated using high gains to reduce the gap between the desired trajectory and the actual trajectory, but they introduce large forces generating potentially dangerous noncompliant movements.[24] In order to prevent high gains, we use an adaptive feedback (AF) controller, which transforms the trajectory error (in sensory coordinates) into a motor command. It self-adapts by a learning rule through consecutive iterations of the same trajectory. In fact, the learnt adapted dynamic arm model (constituted by the LWPR and cerebellar modules) together with the AF controller allows the presented scheme to preform precise and compliant movements even in changing scenarios (for instance, when manipulating different objects).

In the following sections, we will present the advantages of our approach. In the first section, we will describe the recurrent adaptive feedback error learning (RAFEL) control scheme presented in this work, showing its connections and system equations. Then, we will discuss the properties or characteristics of the forward model learning algorithm (LWPR), the cerebellar microcircuit, and how they are integrated in the RAFEL approach. Finally, we will demonstrate the performance of the presented model with experiments on a 3-DOF and a 7-DOF simulated LWR arm.

2. Control System Architecture

The RAFEL block diagram is shown in Fig. 1. The Trajectory planner delivers desired state terms and their derivatives (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$), where d stands for desired. The difference between desired terms (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$) and actual plant state (Q, $\dot{Q}$, $\ddot{Q}$) delivered through the sensory feedback (upper pathway in Fig. 1) is used as error estimates (e_p, e_v, e_a, stating for error in position, in velocity, and in acceleration,

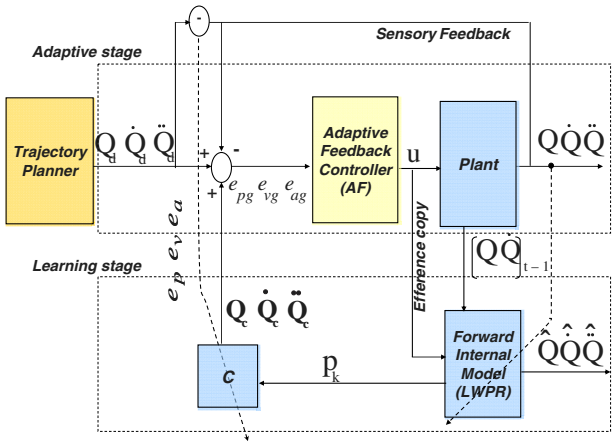

Fig. 1. Block diagram for the RAFEL architecture. Dynamics are encoded in the forward model unit. This unit predicts the next state from the current state of the robot arm and the efference copy or motor command u. The cerebellum module (block C), instead of receiving inputs by means of mossy fibers (MFs), receives preprocessed signals from the LWPR receptive filters (RFs) (p_k) and computes corrections (Q_c, $\dot{Q}_c$, $\ddot{Q}_c$) to be added to desired positions, velocities, and accelerations (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$) (Learning stage). Then, sensory feedback (Q, $\dot{Q}$, $\ddot{Q}$) (i.e. actual position, velocity, and acceleration) is subtracted from the desired terms and the result is sent to the AF controller that generates the motor torque u assuring the stability of the system without using high gains (adaptive stage).

respectively) by the cerebellar module (C) for its internal adaptation learning rule. This cerebellar module delivers corrective terms (Q_c, $\dot{Q}_c$, $\ddot{Q}_c$) in sensory space, to be added to the desired terms (from the planner) and sent towards the plant. The AF controller translates final position, velocity, and acceleration contributions into motor commands. These final torque values (u) are sent to the plant and also (as efferent copy) to the forward internal model (LWPR) of the arm. This forward internal model, implemented with the LWPR, delivers the activity of the p_k filters to the cerebellar module. The cerebellar module takes them as an optimized representation of the plant state. The cerebellar module uses p_k kernel activity as a compact sensory-motor representation. This is the activity that a cerebellum model would have at the granular layer. Besides the cerebellar-based loop, the AF controller contributes to minimize the error, reducing the difference between the sensory feedback and the desired terms (Q_d) on an inner closed loop.

Command torques (u) are computed by the AF controller from the global errors (e_{pg}, e_{vg}, e_{ag}), where g stands for global, according to Eq. (1)

$$u = \hat{M}(Q_{dc})[e_{ag} + K_v e_{vg} + K_p e_{pg} + \hat{B}_r], \quad (1)$$

where $\hat{M}$ is the parametric inertia matrix of the LWR and $\hat{B}_r$ represents the approximated friction force. The derivation of this equation is detailed in Subsec. 2.1. The global errors are obtained according to (2) by summing feedback errors (e_p, e_v, e_a) and sensory corrective terms (Q_c, $\dot{Q}_c$, $\ddot{Q}_c$) which are produced by the simplified cerebellar microcircuit explained in Subsec. 2.2. These terms are called sensorial contribution since they are represented in the sensory space (not in terms of torques).

$$(e_{pg}, e_{vg}, e_{ag}) = (Q_d, \dot{Q}_d, \ddot{Q}_d) - (Q, \dot{Q}, \ddot{Q}) + (Q_c, \dot{Q}_c, \ddot{Q}_c). \quad (2)$$

The cerebellar module produces sensory corrective terms $(Q_c, \dot{Q}_c, \ddot{Q}_c) = C(p_k, e_p, e_v, e_a)$, which are corrections in position, velocity, and acceleration that are estimated by using the kernel activity (p_k) provided by the LWPR module as described in Subsec. 2.2.

As indicated before, the cerebellar circuitry applies corrections for the miscalibration of the system adjusting its weights and taking into account the feedback errors (e_p, e_v, e_a) as described in Subsec. 2.2. From a biological point of view, block C in Fig. 1 consists of a set of uniform cerebellar circuits (single neural layer) which are capable of learning. This layer adaptively modifies its input-output characteristic function in order to reduce the sensory errors (e_p, e_v, e_a). Further details about C are given in Subsec. 2.2. In our approach, the LWPR algorithm is implemented to play the important role of the forward model, which means that it learns the direct dynamic model of the robot arm. The machine learning engine (LWPR) and the cerebellar cortical circuitry (C) complement each other; the former takes advantage of the adaptive cerebellar corrections through the efference copy and the latter, of the efficient reduction of the high dimensional input dimensions, of the incremental learning, and of a compact sensory-motor representation provided by the bank of filters p_k of the LWPR.

The field of nonlinear control theory is very large; we will focus on a particular method called adaptive computed torque control.[33] Considering the analytical robot model, we calculate the joint torques required for a particular trajectory using the dynamic expression of the robot (3):

$$u = M(Q)\ddot{Q} + V(Q, \dot{Q}) + G(Q) + F(Q, \dot{Q}), \quad (3)$$

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, $\dot{Q}$, $\ddot{Q}$ are the joint angles, velocities, and accelerations of the robot arm. In the architecture, the u joint torque values generated by the AF controller allow the robot to follow a desired trajectory (Q_d, $\dot{Q}_d$, $\ddot{Q}_d$) and ensure the stability of the trajectory at the same time. The specifics and equations of the AF controller are presented in Subsec. 2.1.

When the cerebellar output is accurate and the forward dynamic model is exact, the AF controller will cancel the nonlinearities, transforming the resulting global errors (e_{pg}, e_{vg}, e_{ag}) into the correct torque u. Nevertheless, there may be a difference between the desired state and the actual output of the controlled arm; for instance, due to a heavy object being manipulated (affecting the dynamics of the plant+object model) that will activate the cerebellar process of learning.[15]

Summarizing, the cerebellum elaborates the sensory-motor complexes that it receives from the optimized representation provided by the LWPR and produces the sensory corrective terms (Q_c, $\dot{Q}_c$, $\ddot{Q}_c$), which, added to feedback errors, enable the AF controller to adapt the u motor commands. The LWPR engine incrementally learns from the u torque values and the current state of the robot arm $(Q, \dot{Q})_{(t-1)}$ (see Fig. 1) and abstracts the whole forward dynamic model.

2.1. *AF controller*

In this subsection, we address the problem of controlling a robot arm of many DOFs. Taking into account the fact that an analytical computation of the robot dynamics is complex and a large number of parameters may be unknown, we implement an adaptive model for an accurate and stable control during object manipulation: the AF controller. The AF controller ensures the stability of the system producing the appropriate joint torque values to obtain the right execution of the desired trajectory in a compliant way. From Eq. (4), it can be

noticed that feedback joint torque u_r varies after consecutive repetitions of the same trajectory according to the gradual dynamic acquisition process (forward dynamic module) of the moving arm when executing a manipulation task, $r = 0, 1, \ldots$ (r stands for the iteration number).

$$u = \hat{M}(Q_{dc})\ddot{Q}_{dc} + \hat{V}(Q_{dc}, \dot{Q}_{dc}) + \hat{G}(Q_{dc}) + \hat{u}_r, \quad (4)$$

where $(Q_{dc}, \dot{Q}_{dc}, \ddot{Q}_{dc}) = (Q_d, \dot{Q}_d, \ddot{Q}_d) + (Q_c, \dot{Q}_c, \ddot{Q}_c)$. $\hat{M}$, $\hat{V}$, and $\hat{G}$ represent the parametric values of the LWR (see Appendix B). According to the adaptive computed torque theory, we rewrite (4), obtaining Eq. (5).

$$u = \hat{M}(Q_{dc})\tau_p + \tau_c, \quad (5)$$

where $\tau_p = \ddot{Q}_{dc} + K_v\dot{e} + K_p e$, and $\tau_c = \hat{V}(Q_{dc}, \dot{Q}_{dc}) + \hat{G}(Q_{dc}) + \hat{u}_r$. Substituting (5) in (3), we obtain (6) the closed loop control of the system in Fig. 1 (adaptive stage).

$$\ddot{e} + K_v\dot{e} + K_p e = \hat{M}^{-1}[(M - \hat{M})\ddot{Q} + (V - \hat{V}) + (G - \hat{G}) + (F - \hat{u}_r)]. \quad (6)$$

It has been demonstrated that a simple proportional-derivative control is able to ensure stability[34] in traditional manipulators. However, the friction term is crucial in controlling LWR arms (as is our case) with high-ratio gear boxes. In this case, there are no conventional existing methodologies to control these robots without a massive modeling.[35] In order to ensure stability in our system we aim at compensating the friction forces, therefore, Eq. (6) becomes Eq. (7) where the term $\hat{u}_r$ is a nonmodeled friction term added to the other estimated terms. This accounts for the dynamic model which is not well-known or not precise.

$$\ddot{e} + K_v\dot{e} + K_p e = \hat{M}^{-1}(F - \hat{u}_r), \quad (7)$$

or in a more compact form:

$$\ddot{e} + K_v\dot{e} + K_p e = B - B_r, \quad (8)$$

where $\ddot{e} = (\ddot{Q}_{dc} - \ddot{Q})$, $B = \hat{M}^{-1}F$, $B_r = \hat{M}^{-1}\hat{u}_r$. For every joint, Eq. (8) becomes:

$$\ddot{e}_i + K_{vi}\dot{e}_i + K_{pi}e_i = B_i - B_{ir}. \quad (9)$$

In the last expression (9), term B_i is constant during iterations over time, while term B_{ir} changes on consecutive iterations of the task. We choose the learning rule (10) for each joint i:

$$\hat{B}_{i(r+1)} = \hat{B}_{ir} + \omega * e_{ir}, \quad (10)$$

where $\omega * e_{ir}$ is the convolution between the impulse response filter ω and the error in iteration r.

We choose the specific filter defined by the Laplace transform in Eq. (11):

$$\Omega(s) = s^2 + (K_{vi} - \mu)s + (K_{pi} - \mu), \quad (11)$$

where μ, K_{pi}, and K_{vi} are constants. AF controller gains (K_{pi}, K_{vi}) have been set to very low values for a compliant control taking into account the sufficient condition provided by Nakanishi and Schaal in Ref. 36 in order to ensure the stability of the adopted FEL scheme.

Ω is a noncausal filter, so it uses the errors of the previous iterations and its convergence depends on μ. It is worth noting that symbolisms for error values used in this subsection have been slightly modified ($\ddot{e} = e_{ag}$, $\dot{e} = e_{vg}$, $e = e_{pg}$) in order to make the paragraph easier to follow. Further specifics on the AF controller analysis are provided in Appendix A. As indicated in this appendix, the choice of the filter guarantees learning convergence.

2.2. *The role of the forward internal model in the cerebellum*

LWPR works, in the presented architecture, as a model abstraction engine capable of learning the forward dynamic model of the controlled robot arm+object (the object being manipulated).

LWPR is an algorithm for nonlinear function approximation in high dimensional spaces with redundant and irrelevant input dimensions. In order to perform an optimal function approximation, LWPR incrementally divides the input space into a set of K RFs defined by the center c_k and a Gaussian area characterized by the particular kernel width D_k[21] as shown in Eq. (12). Figure 2 represents its processing unit where N inputs enter into K linear local models (these local kernels are incrementally created and allocated).

For each data point x_i, a weight $p(k, i)$ ($i = 1, \ldots, N$, $k = 1, \ldots, K$, N number of inputs, K number of linear local models) is computed inside each local unit k to measure the input locality according to the distance from the center c_k of the kernel. In other words, the weight measures how often an item of data x_i falls into the region of validity of

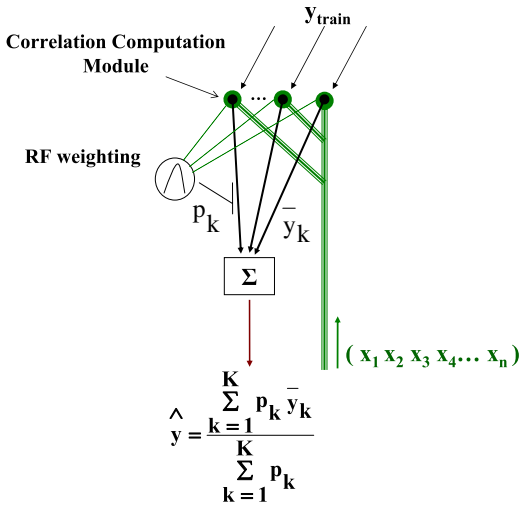

Fig. 2. Schematic layout of the LWPR learning mechanism.

each linear model. The kernel function is defined as a Gaussian kernel:

$$p_k = \exp\left(-\frac{1}{2}(x_i - c_k)^T D_k (x_i - c_k)\right), \quad (12)$$

where D_k is called *distance matrix* and defines the size and shape of the validity region of the linear local model k.

Every time a data sample falls into the validity region of the local model, its distance matrix and regression parameters are updated. This happens independently for each model. In fact, every model makes its own prediction and the combination of all individual predictions $\overline{y}_k$ is the total output $\hat{y}$ (see Eq. (13)) of the LWPR network. In other words, the LWPR prediction is the weighted mean of all linear models:

$$\hat{y} = \frac{\sum_{k=1}^{K} p_k \overline{y}_k}{\sum_{k=1}^{K} p_k}. \quad (13)$$

Forward models predict the next state of each joint (e.g. position, velocity, and acceleration) given the current state and the efference copy.[7,37,38] As a result, LWPR input data consist of torques, current positions, and velocities of all the joints, while the output data are the predictions of the outgoing positions, velocities, and accelerations. In a more detailed way, the LWPR training takes place for each DOF separately. We set up [3 × number of joints] modules, with a test set of [3 × number of joints] inputs (torques, current positions, current velocities) and 1 target. Each LWPR module retains either the next joint position, the next joint velocity, or the next joint acceleration as target signal. The learning goal is to make prediction errors converge to zero, thus providing an optimized representation of the sensory-motor complexes to the cerebellum module.

In our scheme, the cerebellar module (C) takes full advantage of the optimized internal models being continuously updated at the LWPR. In the cerebellum, information coming from the MFs is expansively distributed over many granule cells (GCs) to produce a sparse representation on the parallel fibers (PFs) reaching the PCs. Our cerebellar module only includes this single layer of PCs, that receive an optimized input representation directly from the LWPR (which delivers the activity of its kernels p_k to this Purkinje layer). Therefore kernels p_k play here the role of the MFs→GCs layers. As mentioned before, LWPR optimizes the network size by encoding the input space through the RF kernels. In our set up, the input space (variables x) are joint torque values, current positions, and current velocities. Then, RF kernel functions can be thought as a bank of filters G_k defined in Eq. (14), and the output signals p_k are driven by the PFs and interneurons (by analogy with the cerebellum) to the PCs (explicitly included in our C module).

$$p_k = G_k(x_1, x_2, \ldots, x_i), \quad k = 1, \ldots, K, \quad i = 1, \ldots, N. \quad (14)$$

Then, we take advantage of the LWPR local kernels acting as different granular and molecular layer microzones in the cerebellum model, thus allowing us to obtain a compact sensory-motor representation.

PCs output (which is the cerebellar module output) $z_k(t)$, defined in (15), is modeled as a weighted linear combination of the $p_k(t)$ computed by Eq. (14):

$$z_k(t) = \sum_k w_k p_k(t). \quad (15)$$

The adaptive synaptic weights w_k (in analogy with the PF→PC synapse) are updated using the heterosynaptic covariance learning rule[19] in its continuous form[10] (16) and adjusted by an error signal ($e(t)$).

$$\delta w_k = -\beta e(t) p_k(t), \quad (16)$$

where β is a small positive learning rate and $e(t)$ is the error signal.

In this approach, $e(t)$ is the feedback error signal, which has three components e_p, e_v, and e_a (positions, velocities, and accelerations) for each joint. β is set to 0.005 in all the experiments carried out. The cerebellum produces [3 × number of joints] position, velocity, and acceleration corrections for every joint arm, which are updated by the corresponding weight w_k.

Summarizing, in the LWPR, all RFs calculate their weight activation in order to assign the new input, x_i, to the closest RF and consequently, the center and the kernel width are incrementally updated during the learning process. The optimized choice of centers and widths gives the optimal basis of RFs, so that the accuracy and the learning speed of the cerebellar model are improved. In other words, Eq. (14) represents the bank of filters for the GCs in the cerebellum and they are both used to compute the cerebellar outputs (Q_c, $\dot{Q}_c$, $\ddot{Q}_c$) as a linear weighted combination as defined in Eq. (15) and to update the synaptic weights w_k (16).

Finally, the function to be approximated online by the regression algorithm LWPR during simulation is shown in Eq. (17):

$$\Phi(u, (Q, \dot{Q})_{t-1}) = (Q, \dot{Q}, \ddot{Q})_t, \tag{17}$$

where $(Q, \dot{Q})_{t-1}$ is the current state and $(Q, \dot{Q}, \ddot{Q})_t$ is the next state of the robot that corresponds to the efference copy of the motor command u. Further, implementation details on the approximation process are given in Sec. 3.

3. Materials and Methods

In order to allow the forward internal model (LWPR) to learn different context dynamics, we considered four case-studies when emulating the manipulation of different objects, in which the manipulated object was defined as punctual payloads tied to the simulated robot end effector. We have performed experiments with masses of 2, 6, 8, and 10 Kg. So, we have studied how both the adaptive and learning stages compensate the errors when the simulated robot arm is following a desired trajectory. This combination allows the simulated robot arm to follow the desired trajectory under different contexts (i.e. manipulating different objects). Moreover, we have also tested the stability of the RAFEL architecture under kinematic modifications obtained by shifting the end effector orientation (this aims to emulate the kinematic changes in the robot arm+object when manipulating objects such as a pointer). In other words, we have evaluated the performance of the RAFEL architecture in adapting to dynamics and kinematics changes of the controlled object on two physically realistic models of robotic arms. In the first setup, the LWR arm was simulated considering a reduced configuration to 3-DOFs (fixing the rest of the joints) in order to get fewer input dimensions to the machine learning engine. The three nonfixed joints (Q_1, Q_2, Q_3) used in our experiments are indicated in Fig. 3. This reduces the amount of training data required and expedites the initial learning process. Afterwards, all DOFs available in the DLR LWR III (up to 7), shown in Fig. 3, were involved in the simulation (further information about the LWR simulated model can be seen in Appendix B). LWPR learning was carried out online and the learnt forward internal models were adapted to changes in dynamics at every time step.

For both configurations, the gains of the AF controller have been set to the same value for the four objects and for all the robot joints ($K_p = 6$; $K_v = 3$; $\mu = 0.75$). Nakanishi and Schaal provided in Ref. 36 a strictly positive real (SPR) condition, that is $K_v^2 > K_p$, for choosing feedback gains in order to ensure stability of the feedback error learning mechanism. Our choice of feedback gains satisfies the mentioned condition, which implies that the stability of the RAFEL architecture is guaranteed.

The task for the experiments with the LWR 3-DOF arm was to follow a planned trajectory in

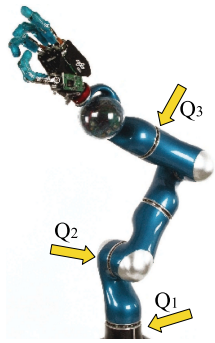

Fig. 3. LWR arm and hand consisting of seven revolute joints. The three joints used in our 3-DOF experiments are explicitly indicated. Figure adapted from Albu-Schäffer *et al.* in Ref. 31.

a three-dimensional task space defined by (18).

$$Q_1 = A\sin(\pi t), \quad Q_2 = A\sin(\pi t + \phi), \quad Q_3 = A\sin(\pi t + 2\phi), \tag{18}$$

where A is a constant (0.1), ϕ is $\pi/4$, and Q_1, Q_2, and Q_3 are the joint coordinates, respectively.

In the second setup, aiming to scale the movement to a 7-DOF scenario, the arm had to follow in Cartesian coordinates the trajectory defined by Eq. (19):

$$y = 0.15\sin(2t), \quad z = 0.6 + 0.2\cos(t), \tag{19}$$

remaining variable x constant.

We ran 25 iterations of the above-stated trajectories and trained online the LWPR in every iteration with 500 points for the first 3-DOF setup and 1000 points for the second 7-DOF setup (this corresponds with half of the points of the whole trajectory in each case). The sampling intervals for the eight-like trajectories were 2 ms and 1 ms, for 3DOF- and 7DOF-robot configurations, respectively. In all the cases, the eight-like trajectory lasted 2 s.

With the purpose of highlighting the importance of the cerebellum in the system and the role of the forward internal model in making more effective cerebellar corrections, we have also compared the performance between the RAFEL architecture with another system configuration in which the cerebellum module has been removed and also with another architecture with high-gains PD instead of the AF controller.

In order to evaluate the functional structure of the LWPR forward internal model within the RAFEL architecture and its capability to generalize among contexts (that can be different conditions, objects being manipulated, etc.), we have performed a generalization experiment. We tested the LWPR capability to provide an optimal basis of RFs to compute the cerebellar corrections under unseen dynamic contexts. Firstly, the LWPR within the RAFEL architecture is trained with some dynamic contexts (2, 6, and 10 kg), secondly, the testing is done with 1, 4, and 8 kg contexts. Furthermore, we tested the performance of the RAFEL architecture in different desired trajectories obtained from Eq. (18) by modifying its coefficients. Experimental results show that the system is able to keep the desired performance thanks to the synergy between cerebellum and AF modules.

We have evaluated the system accuracy considering the performance of the whole iteration of the planned trajectory: the nMSE in radians (Rad) between the desired joint angle Q_d and the actual joint angle Q obtained from the simulated robot arm. The nMSE is defined as the MSE divided by the variance of the target data values.

All simulations were set up in the MATLAB (v.R2008a) environment and we used the robotics toolbox[39] for MATLAB. All experiments carried out for this work and the results obtained are described in Sec. 4.

4. Experiments and Results

With the 3-DOF simulated robot arm, the LWPR creates 22 locally linear models (RFs) for 2–6–8–10 kg contexts. Firstly, we investigated the performance of the control architecture by manipulating four objects (simulated as mass points) at the end effector of the robot arm. Figure 4 shows the average of the nMSE over the three joints; different traces indicate the response to different contexts during 25 trajectory iterations (18). As can be seen, in the three joints, the error is reduced homogeneously.

Figure 4 proves that the RAFEL architecture is adaptable to dynamic transformations in the context of tracing a planned trajectory. In the same way, the averaged nMSE in Fig. 5 shows no remarkable variations in its performance under kinematics transformations of the arm tip. We have obtained these kinematics changes by shifting the orientation of the

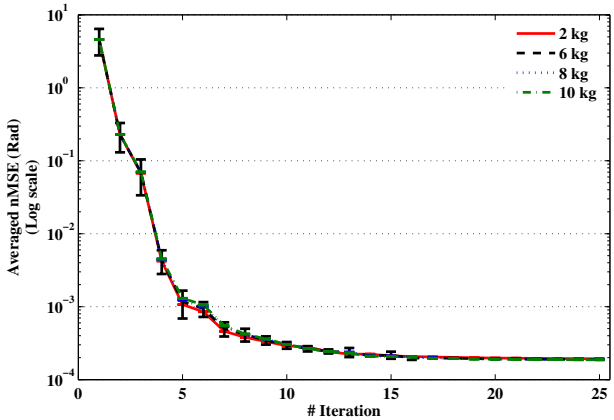

Fig. 4. nMSE averaged over the three joints for four contexts. For the sake of clarity, error bars are plotted only for the 6 kg context and indicate the standard deviation between the joints.

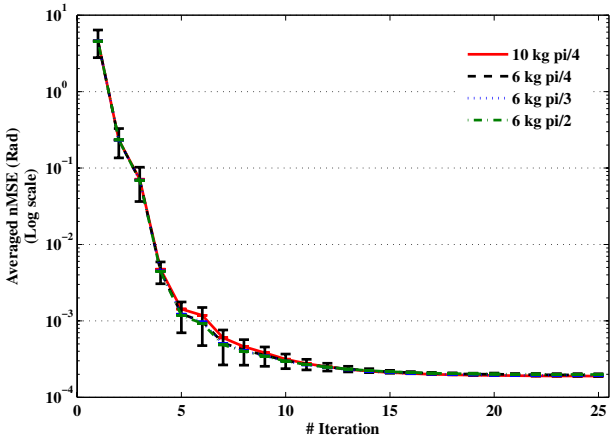

Fig. 5. The lines represent the averaged performance (nMSE) among the three joints during 25 iterations of the desired trajectory. For the sake of clarity, error bars, indicating the standard deviation between the joints, are plotted only for the 6 kg and $\pi/4$ radians context, which is representative of all the other tested contexts.

end effector by a factor $\lambda = [\pi/4, \pi/2]$ in radians. So, the architecture robustness is not affected either by changes in both dynamics or kinematics.

In the previous experiments, the LWPR learnt the contexts separately. Nevertheless, the LWPR is capable of learning different forward dynamics internal models and of retaining them in the same regression model. Figure 6 shows that after training forward dynamics models corresponding to the dynamics of the robot arm manipulating three different payloads (2, 6, and 10 kg) at the arm tip, the

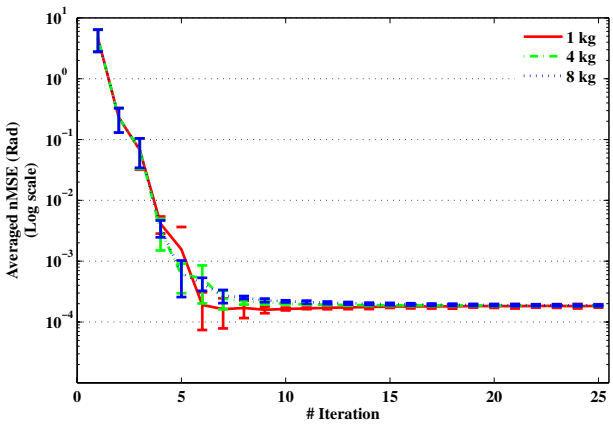

Fig. 6. The RAFEL architecture still shows a good performance when being tested under unseen dynamics contexts. This is expressed by the nMSE value plotted in the figure. Error bars represent the standard deviation between the three joints.

LWPR was tested (switching off the learning mechanisms during the test trials) with three unlearnt payloads of 1, 4, and 8 kg to study its generalization capability to predict the simulated arm behavior under new contexts. The robot was expected to follow the trajectory (18) in all cases.

In order to highlight the advantages of the RAFEL architecture and how its different components complement each other, we examined and compared how the errors, when following a desired trajectory (18), became compensated by using three different architectures. Besides the RAFEL architecture, we considered a second approach with the feedback loop (AF), but without cerebellar corrections (thicker solid line in Fig. 8), and a third one consisting of block C and a high-gain PD instead of the AF controller. These two architectures are drawn in Fig. 7.

In order to guarantee a low error when following a desired trajectory, the system with a PD controller used the gains of the RAFEL architecture multiplied by a factor of 500. But even with this change, the thin solid line in Fig. 8 shows that the final behavior is not stable and the maximum torque applied at joints is 758 Nm, which is a potential risk in case of physical contact of the robot arm with the environment.[40] Figure 8 shows that the RAFEL scheme achieved a better performance even with a maximum torque of around 200 Nm. The influence of mass and velocity of the DLR LWRIII in resulting injuries on human bodies was studied by Haddadin *et al.* in

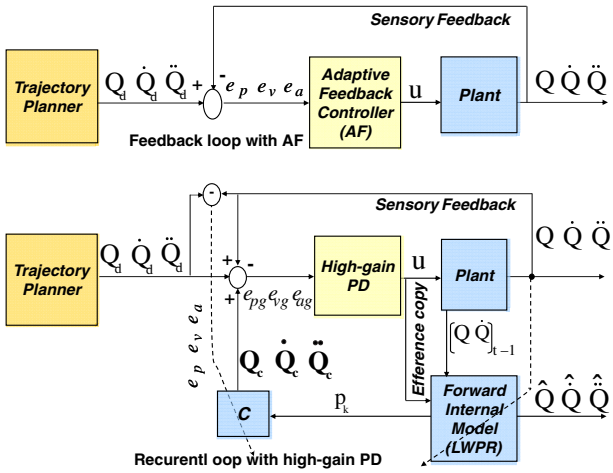

Fig. 7. The feedback loop with AF and the recurrent loop with high-gain PD architectures compared with the RAFEL architecture.

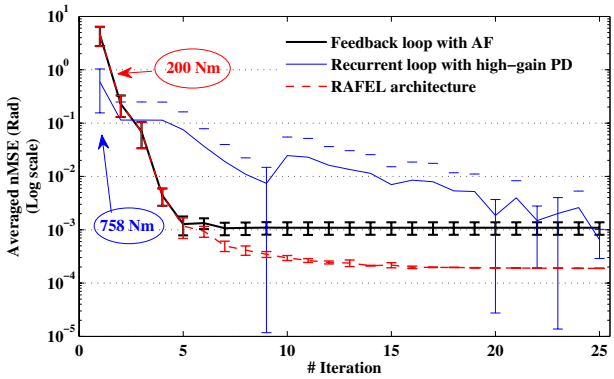

Fig. 8. Performance comparison of three control architectures (see Fig. 7) when a simulated robot arm is manipulating a 6 kg object. Lines display the nMSE averaged over three joints, error bars indicate the standard deviation between the joints. The case of a PD with low gains is not plotted because it leads to a very bad performance (which lays out of range in this plot).

Table 1. Coefficients of the tested trajectories.

	A	ϕ
Traj. 1	0.1	$\pi/4$
Traj. 2	0.1	$\pi/2$
Traj. 3	0.05	$\pi/4$
Traj. 4	Traj. 1 + Traj. 3	

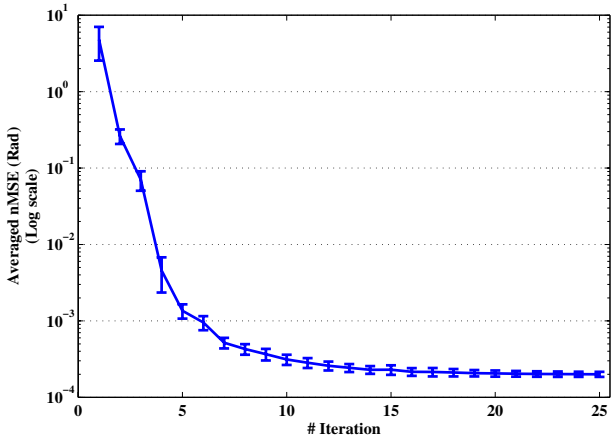

Fig. 9. The RAFEL architecture shows a good performance during the test stage with new trajectories whose coefficients are defined in Table 1. The robot arm manipulated a 6 kg load. The line indicates the mean nMSE value, averaged over three joints firstly for each trajectory and then, over the four trajectories. Error bars represent the standard deviation above and below the mean of the nMSE of the four trajectories.

Ref. 40. In these tests, it becomes clear that arbitrary masses moving at speeds below 2 m/s were not able to become dangerous to a nonclamped human head. Furthermore, they indicated that the inertia properties of the LWRIII allow impact velocities of up to 1 m/s without leading to soft-tissue injuries. With regard to our system, the maximum linear velocity obtained during the self-adaptive experiment for the 3-DOF LWR arm manipulating 10 kg load at the last joint is 0.81 m/s; therefore, it can be seen as a compliant approach. This result has been computed considering the worst case, i.e. the longest link and the maximum angular velocity.

We also tested other trajectories obtained from Eq. (18) by changing the phase or the amplitude or

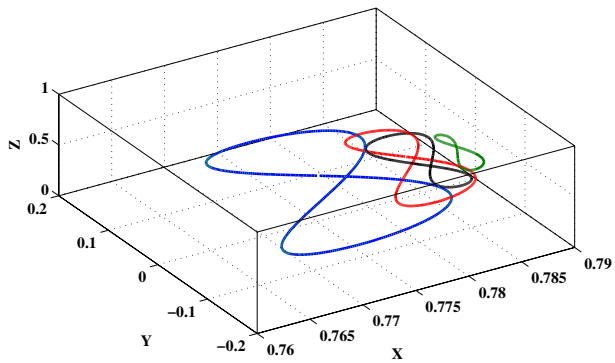

Fig. 10. The eight-like figure-shapes obtained after 25 iterations for the four precomputed trajectories (they are accurately approximating the desired trajectories).

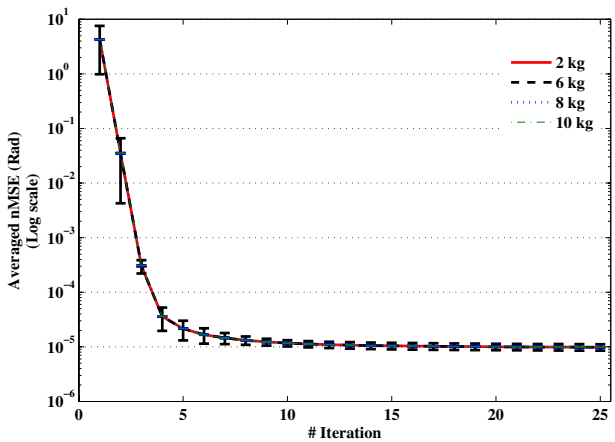

Fig. 11. The RAFEL architecture scales with a good performance using a simulated robot arm of 7-DOFs when its dynamics are modified by manipulating different payloads.

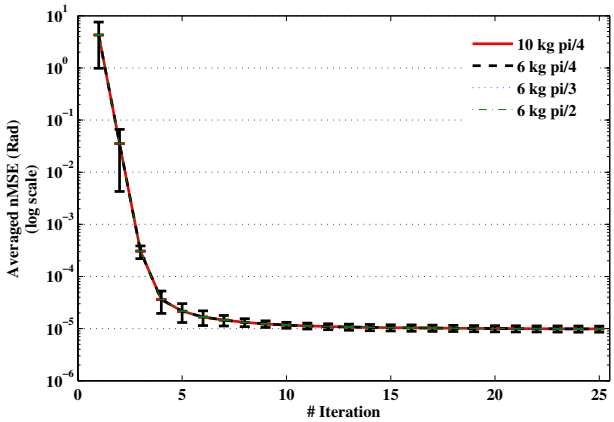

Fig. 12. The RAFEL architecture also scales to 7-DOFs with a good performance under kinematics transformations.

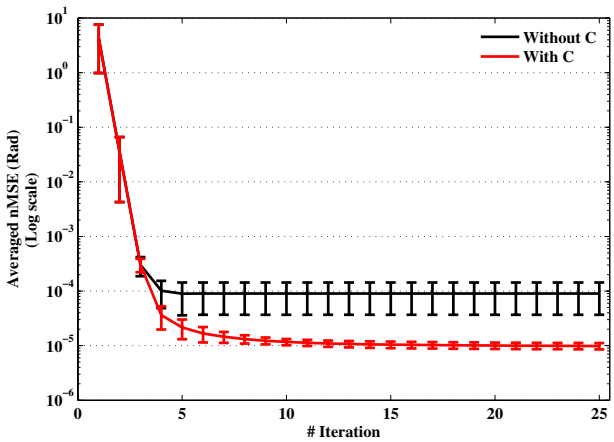

Fig. 13. Lines represent the nMSE averaged over 7 joints after having been averaged over the four dynamic contexts (2, 6, 8, and 10 kg). The solid black line shows the tracking error performance obtained by removing the cerebellar block in the RAFEL scheme. Comparing them, it becomes clear that the cerebellum drives the system to achieve a better performance in terms of nMSE with a small deviation among joints. Error bars represent the standard deviation above and below the mean of the nMSE of the seven joints.

summing both of them. For these experiments, we composed four trajectories with the coefficients indicated in Table 1. During the experiments, half of the points along each trajectory have been used for learning and the other half, for testing. The performance results of tracking these trajectories when manipulating a 6 kg payload are shown in Fig. 9 and their eight-like figure-shapes are plotted in Fig. 10, which corresponds to the final iteration number 25.

The RAFEL architecture has been also tested with a higher number of DOFs and results in Figs. 11 and 12 indicate that the system's behavior becomes stable after a few iterations of the desired trajectory (19). The LWPR creates 25 locally linear models (RFs) for each learnt context. Aiming to highlight the important role of the cerebellum in driving all the joints to converge at a similar nMSE error value, we compared the performance of the RAFEL architecture with a similar one without the C block applying sensory corrections to the desired trajectory. Results are shown in Fig. 13.

5. Conclusions

In this paper, we have proposed an adaptive and predictive control architecture based on different elements: a cerebellar module (C) which provides corrective terms (in position, velocity, and acceleration corrections), a machine learning module (LWPR) which incrementally acquires an optimized representation of the input sensory-motor space (this is then used as input to the cerebellar module), and an adaptive AF module which translates position, velocity, and acceleration into actual torque values but using low gains. The combination of these different elements on the same architecture delivers a compliant control in a simulated robot arm in manipulation tasks. In the performed experiments, it is shown that the manipulated objects effectively affected the dynamics or kinematics of the robot+object model; however, the proposed approach provided a compliant control (using low torque values) with high accuracy since the internal models were efficiently adapted towards compensating the mismatch between the base model (only robot arm) and the modifications in each context (robot+object under manipulation). Furthermore, the RAFEL architecture leads the joints to quickly converge to a small position error taking advantages of the LWPR's capability to retain and generalize different dynamics and kinematics contexts.

In the proposed scheme, the input layer of the cerebellum model was abstracted efficiently by using the LWPR internal kernels. The experiments done in this study also show how an efficient and continuously adapting input sensory-motor space representation can efficiently be used by a simple integrative neural layer (PCs) in the framework of a recurrent

control loop. The importance of an efficient and clean contribution to the Purkinje layer has been recently evaluated[41,42] and other authors, such as Porrill and Dean in Ref. 10 and Schweighofer *et al.* in Ref. 43 hypothesized that the cerebellar learning is facilitated by optimizing the choice of the centers and the basis of RFs at the granular layer. We have also illustrated the complementary role of the LWPR and the cerebellar modules with specific experiments. Here, the RAFEL architecture was compared with other approaches in which the cerebellar module or the LWPR modules had been removed and substituted by other components such as high-gain PD.

The proposed scheme is based on error estimates in the sensory space (position, velocity, and acceleration). This recurrent loop avoids dealing with the *distal error* problem and provides an efficient motor control.

As future work, we suggest studying the exploitation of the learnt forward internal models to provide an estimate of the outcome of a motor command. Then, we could combine the recurrent control loop for compensating deviations in the trajectory and forward model predictions.

Acknowledgments

This work has been supported by the EU projects SENSOPAC (IST-028056), REALNET (FP7-270434), and by the Sardinian Master and Back Program (P.O.R. FSE 2007–2013).

Appendix A

Substituting the Laplace transform of Eq. (9) in the Laplace transform of Eq. (10), we obtain Eq. (A.1):

$$\hat{B}_{i(r+1)}(s) = \hat{B}_{ir}(s) + \Omega(s)H(s) \times [B_i(s) - \hat{B}_{ir}(s)], \quad \text{(A.1)}$$

where

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

By substituting expression (A.3) in Eq. (A.1) we obtain Eq. (A.4):

$$\Gamma(s) = 1 - \Omega(s)H(s), \quad \text{(A.3)}$$

$$\hat{B}_{i(r+1)}(s) = \Gamma(s)\hat{B}_{ir}(s) + B_i(s)[1 - \Gamma(s)]. \quad \text{(A.4)}$$

Starting with $\hat{B}_{i0}$, after r iterations we get Eq. (A.5):

$$\hat{B}_{ir}(s) = B_i(s) + A\Gamma^r(s), \quad \text{(A.5)}$$

where A is a constant.

The convergence of the learning algorithm depends on factor $\Gamma^r(s)$ in Eq. (A.5) and will occur if $\Gamma^r(s)$ approaches zero. In this case, $\hat{B}_{i(r)}(s) \to B_i(s)$, and Eq. (8) will be true with its right side equal to zero.

The inverse Laplace transform of $\Gamma^r(s)$ is defined by Eq. (A.6).

$$\gamma_r(t) = L^{-1}[\Gamma^r(s)]. \quad \text{(A.6)}$$

It has been demonstrated in Ref. 44 that choosing an appropriate filter $\Omega(s)$ as the one in Eq. (11), we would obtain

$$\lim_{r\to\infty} |g_r(t)| = 0, \quad \text{(A.7)}$$

with which it is possible to guarantee the convergence. As indicated in the main text, we have chosen a filter $\Omega(s)$ that fulfills this convergence condition.

Appendix B

The approximate dynamic equation defining the LWR robot is given by the expression:

$$u = M(Q)\ddot{Q} + V(Q, \dot{Q}) + G(Q), \quad \text{(B.1)}$$

Table B.1. Inertia tensor parameters ($\text{kg} \cdot \text{m}^2$).

Joint j	xx_j	xy_j	xz_j	yy_j	yz_j	zz_j
Joint $j = 1$	0.0216417	0.0	0.0	0.0214810	0.0022034	0.0049639
Joint $j = 2$	0.0244442	0.0	0.0	0.0052508	0.0036944	0.0239951
Joint $j = 3$	0.0213026	0.0	0.0	0.0210353	0.0022204	0.0046970
Joint $j = 4$	0.0231668	0.0	0.0	0.0048331	0.0034937	0.0227509
Joint $j = 5$	0.0081391	0.0	0.0	0.0075015	0.0021299	0.0030151
Joint $j = 6$	0.0033636	0.0	0.0	0.0029876	0.0	0.0029705
Joint $j = 7$	0.0000793	0.0	0.0	0.0000783	0.0	0.0001203

Table B.2. Center of mass, and motor inertia (Units: m, kg, and $\mathrm{kg} \cdot \mathrm{m}^2$).

Joint j	mx_j	my_j	mz_j	m_j	I_{mj}
Joint $j = 1$	0.0	0.01698	−0.05913	2.7082	415.50e-6
Joint $j = 2$	0.0	0.11090	0.01410	2.7100	415.50e-6
Joint $j = 3$	0.0	−0.01628	−0.06621	2.5374	361.60e-6
Joint $j = 4$	0.0	−0.10538	0.01525	2.5052	138.50e-6
Joint $j = 5$	0.0	0.01566	−0.12511	1.3028	54.10e-6
Joint $j = 6$	0.0	0.00283	−0.00228	1.5686	60.08-6
Joint $j = 7$	0.0	0.0	0.06031	0.1943	60.08e-6

Table B.3. Friction parameter values: Dry friction and Viscous friction (units: $N \cdot \mathrm{m}$ and $N \cdot \mathrm{m} \cdot \mathrm{s/rad}$).

Joint j	$\pm F_d(Q, \dot{Q})$	$F_v(Q, \dot{Q})$
Joint $j = 1$	±0.35	2.0e-3
Joint $j = 2$	±0.35	1.698e-3
Joint $j = 3$	±0.35	1.660e-3
Joint $j = 4$	±0.35	2.400e-3
Joint $j = 5$	±0.35	1.800e-3
Joint $j = 6$	±0.35	1.200e-3
Joint $j = 7$	±0.35	1.200e-3

where u is the applied torque, $M(Q)$ is the inertia matrix (symmetric positive definite matrix), $V(Q, \dot{Q})$ is the Coriolis and centrifugal force matrix, and finally, $G(Q)$ is the gravitational force vector. The position in joint coordinates is given by Q, the joint velocity by $\dot{Q}$, and the joint acceleration by $\ddot{Q}$. Both dry friction and viscous friction forces are added to the previous equation (B.1) obtaining:

$$u = M(Q)\ddot{Q} + V(Q, \dot{Q}) + G(Q) + F_d(Q, \dot{Q}) \pm F_v(Q, \dot{Q}), \quad \text{(B.2)}$$

where $F_d(Q, \dot{Q})$ and $F_v(Q, \dot{Q})$ are the modeled dry/viscous friction matrices.

The first three terms of Eq. (B.2) mainly include the inherent robot dynamic parameters (inertia matrix, Coriolis matrix, and gravitational force vector). These parameters are up to 11 per joint (inertia matrix is symmetrical):

(1) Inertia tensor terms (xx_j, xy_j, xz_j, yy_j, yz_j, zz_j where $j = \{1, 2, \ldots \text{number of joints}\}$)
(2) Center of mass (mx_j, my_j, mz_j where $j = \{1, 2, \ldots \text{number of joints}\}$)
(3) Mass (m_j where $j = \{1, 2, \ldots \text{number of joints}\}$)
(4) Motor inertia (I_{mj} where $j = \{1, 2, \ldots \text{number of joints}\}$)

These parameters are usually grouped according to these four categories, needing to make the computational task easier.[45] The length of the three links that constitute the robot arm is $L = \{0.310, 0.4 \text{ and } 0.390\,\mathrm{m}\}$. For our particular simulated LWR robot,[31] the nominal values obtained applying parametric methods[46] are shown in Tables B.1–B.3.

References

1. D. Marr, A theory of cerebellar cortex, *J. Physiol.* **202** (1969) 437–470.
2. J. S. Albus, A theory of cerebellar function, *Math. Biosciences* **10**(1–2) (1971) 25–61.
3. M. Ito, Historical review of the significance of the cerebellum and the role of Purkinje cells in motor learning, *Ann. New York Acad. Sci.* **978** (2002) 273–288.
4. S. Ghosh-Dastidar and H. Adeli, Spiking neural networks, *Int. J. Neural Syst.* **19**(4) (2009) 295–308.
5. M. Kawato, Feedback-error-learning neural network for supervised motor learning, in *Advanced*

Neural Computers, ed. R. Eckmiller (Elsevier, North-Holland, 1990), pp. 365–372.
6. H. Gomi and M. Kawato, Adaptive feedback control models of the vestibulocerebellum and spinocerebellum, *Biol. Cybern.* **68**(2) (1992) 105–114.
7. D. M. Wolpert, R. C. Miall and M. Kawato, Internal models in the cerebellum, *Trends Cogn. Sci.* **2**(9) (1998) 338–347.
8. N. R. Luque, J. A. Garrido, R. R. Carrillo, S. Tolu and E. Ros, Adaptive cerebellar spiking model embedded in the control loop: Context switching and robustness against noise, *Int. J. Neural Syst.* **21**(5) (2011) 385–401.
9. M. I. Jordan and D. E. Rumelhart, Forward models: Supervised learning with a distal teacher, *Cogn. Sci.* **16** (1992) 307–354.
10. J. Porrill and P. Dean, Recurrent cerebellar loops simplify adaptive control of redundant and nonlinear motor systems, *Neural Comput.* **19**(1) (2007) 170–193.
11. M. Fujita, Adaptive filter model of the cerebellum, *Biol. Cybern.* **45** (1982) 195–206.
12. P. Dean, J. Porrill, C. Ekerot and H. Jörntell, The cerebellar microcircuit as an adaptive filter: Experimental and computational evidence, *Nat. Rev. Neurosci.* **11**(1) (2010) 30–43.
13. M. Kawato Internal models for motor control and trajectory planning, *Curr. Opin. Neurobiol.* **9**(6) (1999) 718–723.
14. V. Mohan and P. Morasso, Towards reasoning and coordinating action in the mental space, *Int. J. Neural Syst.* **4**(17) (2007) 329–341.
15. M. Ito, Control of mental activities by internal models in the cerebellum, *Nature Rev. Neurosci.* **9**(4) (2008) 304–313.
16. S. Schaal and N. Schweighofer, Computational motor control in humans and robots, *Curr. Opin. Neurobiol.* **15**(6) (2005) 675–82.
17. D. M. Wolpert, Z. Ghahramani and M. I. Jordan, An internal model for sensorimotor integration, *Science* **269** (1995) 1880–1882.
18. R. C. Miall and D. M. Wolpert, Forward models for physiological motor control, *Neural Netw.* **9** (1996) 1265–1279.
19. T. J. Sejnowski, Storing covariance with nonlinearly interacting neurons, *J. Math. Biol.* **4**(4) (1977) 303–321.
20. S. Vijayakumar and S. Schaal, Locally weighted projection regression: Incremental real time learning in high dimensional space, in *Proc. 17th Int. Conf. on Machine Learning* (*ICML'00*) (San Francisco, 2000), pp. 1079–1086.
21. S. Vijayakumar, A. D'Souza and S. Schaal, Incremental online learning in high dimensions, *Neural Comput.* **17** (2005) 2602–2634.
22. C. G. Atkeson, J. G. Hale, F. Pollick, M. Riley, S. Kotosaka, S. Schaul, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, E. Kawato and M. Kawato, Using humanoid robots to study human behavior, *Intell. Syst. Appl. IEEE* **15**(4) (2000) 46–56.
23. S. Schaal, C. G. Atkeson and S. Vijayakumar, Scalable techniques from nonparametric statistics for real time robot learning, *Appl. Intell.* **17**(1) (2002) 49–60.
24. D. Nguyen-Tuong and J. Peters, Learning robot dynamics for computed torque control using local Gaussian processes regression, in *Proc. ECSIS Symp. Learning and Adaptive Behaviour in Robotic Systems* (*LAB-RS'08*) (2008).
25. C. K. I. Williams and C. E. Rasmussen, Gaussian processes for regression, in *Advances in Neural Information Processing Systems*, Vol. 8 (MIT press, 1996), pp. 514–520.
26. A. J. Smola and B. Schölkopf, A tutorial on support vector regression, *Stat. Comput.* **14**(3) (2004) 199–222.
27. G. Lebrun, C. Charrier, O. Lezoray and H. Cardot, Tabu search model selection for SVM, *Int. J. Neural Syst.* **18**(1) (2008) 19–31.
28. O. Sigaud, C. Salaun and V. Padois, On-line regression algorithms for learning mechanical models of robots: A survey, *Robot. Auton. Syst.* **59**(12) (2011) 1115–1129.
29. D. Mitrovic, S. Klanke and S. Vijayakumar, Adaptive optimal control for redundantly actuated arms, in *Proc. 10th Int. Conf. on Simulation of Adaptive Behavior*: *From Animals to Animats* (*SAB'08*) (2008) 93–102.
30. G. Hirzinger, J. Butterfaß, M. Fischer, M. Grebenstein, M. Hähnle, H. Liu, I. Schäfer and N. Sporer, A mechatronics approach to the design of light-weight arms and multifingered hands, in *Proc. ICRA* (2000), pp. 46–54.
31. A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck and G. Hirzinger, The DLR lightweight robot: Design and control concepts for robots in human environments, *Ind. Robot* **34**(5) (2007) 376–385.
32. P. Van der Smagt, Cerebellar control of robot arms, *Connect. Sci.* **10** (January 1998) 301–320.
33. R. H. Middleton and G. C. Goodwin, Adaptive computed torque control for rigid link manipulators, *Syst. Control Lett.* **10**(1) (1988) 9–16.
34. S. Arimoto and F. Miyazaki, Stability and robustness of PID feedback control for robot manipulators of sensory capability, in *Robotics Research*: *First Int. Symp.*, eds. M. Brady and R. P. Paul (1984).
35. P. van der Smagt, Benchmarking cerebellar control, *Robot. Auton. Syst.*, **32** (2000) 237–51.
36. J. Nakanishi and S. Schaal, Feedback error learning and nonlinear adaptive control, *Neural Netw.* **17**(10) (2004) 1453–1465.
37. D. M. Wolpert, Computational approaches to motor control, *Trends Cogn. Sci.* **1**(6) (1997) 209–216.

38. P. R. Davidson and D. M. Wolpert, Widespread access to predictive models in the motor system: A short review, *J Neural Eng.* **2**(3) (2005) 313–319.
39. P. Corke, A robotics toolbox for matlab, *IEEE Robot. Autom. Mag.* **3**(1) (1996) 24-32.
40. S. Haddadin, A. Albu-Schäffer and G. Hirzinger, Safe physical Human-Robot interaction: Measurements, analysis and new insights, in *Springer Tracts in Advanced Robotics* (*ISRR*), eds. M. Kaneko and Y. Nakamura, Vol. 66 (Springer, 2007), pp. 395–407.
41. P. Wulff, M. Schonewille, M. Renzi, L. Viltono, M. Sasso-Pognetto, A. Badura, Z. Gao, F. E. Hoebeek, S. van Dorp, W. Wisden, M. Farrant and C. I. De Zeeuw, Synaptic inhibition of Purkinje cells mediates consolidation of vestibulo-cerebellar motor learning, *Nature Neurosci.* **12**(8) (2009) 1042–1049.
42. T. Honda, T. Yamazaki, S. Tanaka and T. Nishino, A possible mechanism for controlling timing representation in the cerebellar cortex, in *Int. Symp. Neural Networks* (2010), pp. 67–76.
43. N. Schweighofer, K. Doya and F. Lay, Unsupervised learning of granule cell sparse codes enhances cerebellar adaptive control, *Neurosci.* **103** (2001) 35–50.
44. J. J. Craig, *Introduction to Robotics*: *Mechanics and Control*, 3rd edn. (Pearson/Prentice Hall, Upper Saddle River, New Jersey, 2005).
45. W. Khalil and E. Dombre, *Modeling Identification and Control of Robots* (Hermes Penton Science, 2002).
46. B. Bona and A. Curatella, Identification of industrial Robot Parameters for Advanced Model-Based Controllers Design, in: *ICRA* (2005), pp. 1693–1698.