Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2012, Article ID 761019, 11 pages
doi:10.1155/2012/761019

Research Article

Dynamics Model Abstraction Scheme Using Radial Basis Functions

Silvia Tolu,[1] Mauricio Vanegas,[2] Rodrigo Agís,[1] Richard Carrillo,[1] and Antonio Cañas[1]

[1] *Department of Computer Architecture and Technology, CITIC ETSI Informática y de Telecomunicación, University of Granada, Spain*
[2] *PSPC Group, Department of Biophysical and Electronic Engineering (DIBE), University of Genoa, Italy*

Correspondence should be addressed to Silvia Tolu, stolu@atc.ugr.es

Received 27 July 2011; Accepted 24 January 2012

Academic Editor: Wen Yu

This paper presents a control model for object manipulation. Properties of objects and environmental conditions influence the motor control and learning. System dynamics depend on an unobserved external context, for example, work load of a robot manipulator. The dynamics of a robot arm change as it manipulates objects with different physical properties, for example, the mass, shape, or mass distribution. We address active sensing strategies to acquire object dynamical models with a radial basis function neural network (RBF). Experiments are done using a real robot's arm, and trajectory data are gathered during various trials manipulating different objects. Biped robots do not have high force joint servos and the control system hardly compensates all the inertia variation of the adjacent joints and disturbance torque on dynamic gait control. In order to achieve smoother control and lead to more reliable sensorimotor complexes, we evaluate and compare a sparse velocity-driven versus a dense position-driven control scheme.

1. Introduction

Current research of biomorphic robots can highly benefit from simulations of advance learning paradigms [1–5] and also from knowledge acquired from biological systems [6–8]. The basic control of robot actuators is usually implemented by adopting a classic scheme [9–14]: (a) the desired trajectory, in joint coordinates, is obtained using the inverse kinematics [12, 14, 15] and (b) efficient motor commands drive, for example, an arm, making use of the inverse dynamics model or using high power motors to achieve rigid movements. Different controlling schemes using hybrid position/velocity and forces have been introduced mainly for industrial robots [9, 10]. In fact, industrial robots are equipped with digital controllers, generally of PID type with no possibility of modifying the control algorithms to improve their performance. Robust control (control using PID paradigms [12, 14–19]) is very power consuming and highly reduces autonomy of nonrigid robots. Traditionally, the major application of industrial robots is related to tasks that require only position control of the arm. Nevertheless, there are other important robotic tasks that require interaction between the robot's end-effector and the environment. System dynamics depend on an unobserved external context [3, 20], for example, work load of a robot manipulator.

Biped robots do not have high force joint servos and the control system hardly compensates all the inertia variation of the adjacent joints and disturbance torque on dynamic gait control [21]. Motivated by these concerns and the promising preliminary results [22], we evaluate a sparse velocity-driven control scheme. In this case, smooth motion is naturally achieved, cutting down jerky movements and reducing the propagation of vibrations along the whole platform. Including the dynamic model into the control scheme becomes important for an accurate manipulation, therefore, it is crucial to study strategies to acquire it. There are other bioinspired model abstraction approaches that could take advantage of this strategy [23–26]. Conventional artificial neural networks [27, 28] have been also applied to this issue [29–31]. In biological systems [6–8], the cerebellum [32, 33] seems to play a crucial role on model extraction tasks during manipulation [12, 34–36]. The cerebellar cortex has a unique, massively parallel modular architecture [34, 37, 38] that appears to be efficient in the model abstraction [11, 35].

Human sensorimotor recognition [8, 12, 37] continually learns from current and past sensory experiences. We set up an experimental methodology using a biped robot's arm [34, 36] which has been equipped, at its last arm-limb, with three acceleration sensors [39] to capture the dynamics of the different movements along the three spatial dimensions. In human body, there are skin sensors specifically sensitive to acceleration [8, 40]. They represent haptic sensors and provide acceleration signals during the arm motion. In order to be able to abstract a model from object manipulation, accurate data of the movements are required. We acquire the position and the acceleration along desired trajectories when manipulating different objects. We feed these data into the radial basis function network (RBF) [27, 31]. Learning is done off-line through the knowledge acquisition module. The RBF learns the dynamic model, that is, it learns to react by means of the acceleration in response to specific input forces and to reduce the effects of the uncertainty and nonlinearity of the system dynamics [41, 42].

To sum up, we avoid adopting a classic regular point-to-point strategy (see Figure 1(b)) with fine PID control modules [11, 14, 15] that drive the movement along a finely defined trajectory (position-based). This control scheme requires high power motors in order to achieve rigid movements (when manipulating heavy objects) and the whole robot augments vibration artefacts that make difficult accurate control. Furthermore, these platform jerks induce high noise in the embodied accelerometers. Contrary to this scheme, we define the trajectory by a reduced set of target points (Figure 1(c)) and implement a velocity-driven control strategy that highly reduces jerks.

2. Control Scheme

The motor-driver controller and the communication interface are implemented on an FPGA (Spartan XC31500 [43] embedded on the robot) (Figure 1(a)). Robonova-I [44] is a fully articulating mechanical biped robot, 0.3 m, 1.3 kg, controlled with sixteen motors [44]. We have tested two control strategies for a specific "L" trajectory previously defined for the robot's arm controlled with three motors (one in the shoulder and two in the arm, see Figure 2. This whole movement along three joints makes use of three degrees of freedom during the trajectory. The movement that involves the motor at the last arm-limb is the dominant one, as is evidenced from the sensorimotor values in Figure 4(b).

2.1. Control Strategies. The trajectory is defined in different ways for the two control strategies.

2.1.1. Dense Position-Driven. The desired joint trajectory is finely defined by a set of target positions (P_i) that regularly sample the desired movement (see Figure 1(b)).

2.1.2. Sparse Velocity-Driven. In this control strategy, the joint trajectory definition is carried out by specifying only positions related to changes in movement direction.

During all straight intervals, between a starting position P_1 and the next change of direction P_{cd}, we proceed to command the arm by modulating the velocity towards P_{cd} (see Figure 1(c)). Velocities (V_n) are calculated taking into account the last captured position and the time step in which the position has been acquired. Each target velocity for period T is calculated with the following expression, $V_n = (P_{\text{cd}} - P_n)/T$. The control scheme applies a force proportional to the target velocity. Figure 1(d(B)) illustrates how a smoother movement is achieved using sparse velocity-driven control.

The dense position-driven strategy reveals noisy vibrations and jerks because each motor always tries to get the desired position in the minimum time, whereas it would be better to efficiently adjust the velocity according to the whole target trajectory. The second strategy defines velocities dynamically along the trajectory as described above. Therefore, we need to sample the position regularly to adapt the velocity.

2.2. Modelling Robot Dynamics. The dynamics of a robot arm have a linear relationship to the inertial properties of the manipulator joints [45, 46]. In other words, for a specific context r they can be written in the form (1),

$$\tau = \Upsilon(q, \dot{q}, \ddot{q}) \cdot \pi_r, \tag{1}$$

where q, $\dot{q}$, and $\ddot{q}$ are joint angles, velocities, and accelerations respectively. Equation (1), based on fundamentals of robot dynamics [47], splits the dynamics in two terms. $\Upsilon(q, \dot{q}, \ddot{q})$ is a term that depends on kinematics properties of the arm such as direction of the axis of rotation of joints, and link lengths. Term π_r is a high-dimensional vector containing all inertial parameters of all links of the arm [45]. Now, let us consider that the manipulated objects are attached at the end of the arm, so we model the dynamics of the arm as the manipulated object being the last link of the arm. Then, manipulating different objects is equivalent to changing the physical properties of the last link of the arm. The first term of the equation remains constant in different models. It means that all kinematic quantities of the arm remain the same between different contexts. Under this assumption, we could use a set of models with known inertial parameters to infer a predictive model (RBF) of dynamics [27, 31, 48] for any possible context. From another point of view, the previous dynamic Equation (1) could be written in the form (2):

$$\tau = A(q)(\ddot{q}) + H(q, \dot{q}). \tag{2}$$

Each dynamic term is nonlinear and coupled [49] where $A(q)$ is the matrix of inertia, symmetric, and positive and $H(q, \dot{q})$ includes Coriolis, gravitational, and centrifugal forces. $A(q)$ could be inverted for each robot configuration. Approximating $A(q) = \widehat{A}(q)$ and $H(q, \dot{q}) = \widehat{H}(q, \dot{q})$, we obtain (3):

$$\widehat{\tau} = \widehat{A}(q) \cdot u + \widehat{H}(q, \dot{q}), \tag{3}$$

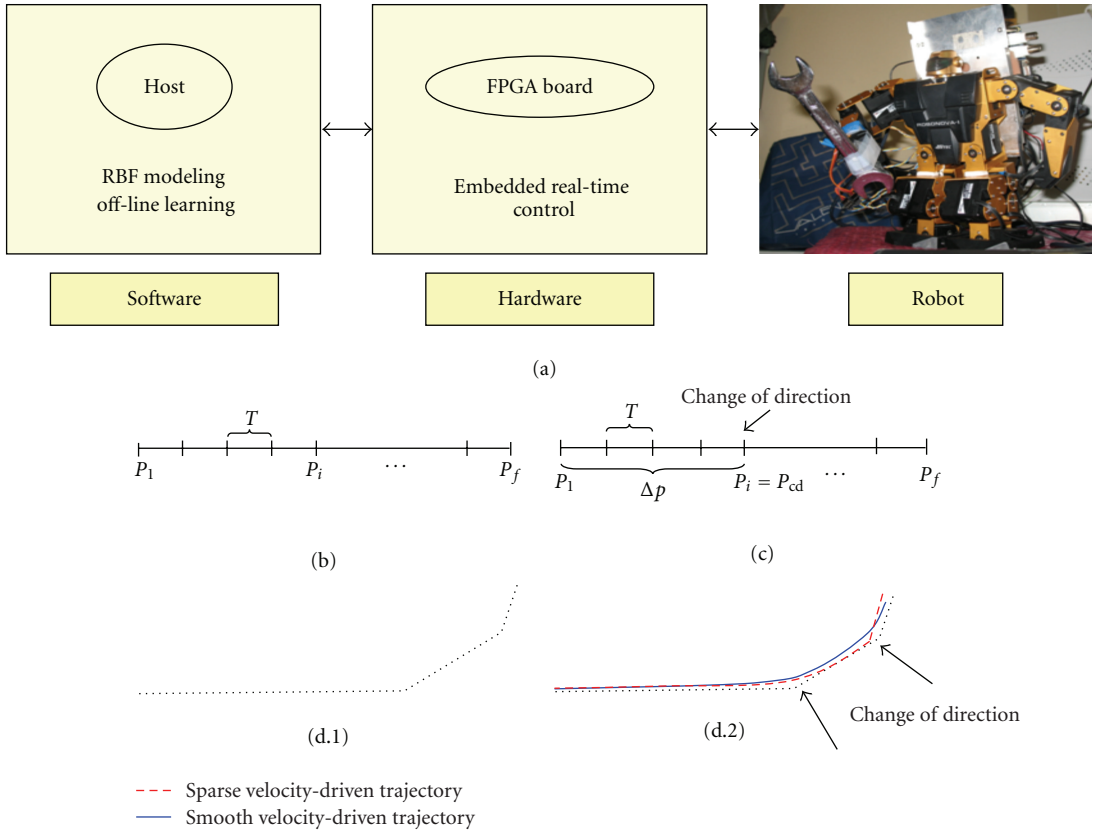

FIGURE 1: (a) Computing scheme. An FPGA processing platform embeds all the real-time control interfaces. The robot is connected to the FPGA which receives commands from the host PC. The robot is equipped with three accelerometers in its arm. The computer captures the sensor data. A RBF is used for learning the objects models from sensorimotor complexes. (b) Dense position-driven scheme. A motor moves from an initial position P_1 to a final position P_f temporally targeting each intermediate point P_i along this trajectory. (c) Sparse velocity-driven scheme. For each position P_i indicated above, a target velocity is calculated for a motor with a time step T of 0.015 s. Using a fixed sampling frequency for obtaining feedback position, we calculate the distance ΔP from this current position to a position P_{cd} corresponding to a change of direction. (d(A)) Dense position-driven trajectory. The trajectory is defined in terms of regularly sampled intermediate points (dashed line). (d(B)) The thinner continuous line represents the real curve executed by the robot's arm in the space under the sparse velocity-driven control. The black arrows indicate points in which changes of direction take place. The thicker continuous line corresponds to the smooth velocity-driven trajectory executed anticipating the points in which changes of direction are applied before the arm reaches them.

where u is a new control vector. $\hat{A}(q)$ and $\hat{H}(q, \dot{q})$ are estimations of the real robot terms. So we have the following equivalence relation (4):

$$u = \ddot{q}. \tag{4}$$

Component u is influenced only by a second-order term, it depends on the joint variables $(q, \dot{q})$, independently from the movement of each joint. To sum up, we replaced the nonlinear and coupled dynamic expressions with a second-order linear system of noncoupled equations. The inputs to the RBF network are the positions and velocities of the robot joints, while the outputs are the accelerations. The input-output relationship is in accordance with the equations of motion. In this work, we propose and show the RBF to be useful in approximating the unknown nonlinearities of the dynamical systems [41, 42, 50–56] through the control signal (4) found for the estimation method for the dynamics of the joints of a robot.

2.3. The RBF-Based Modelling. The RBF function is defined as (5):

$$z(x) = \phi(||x - \mu||), \tag{5}$$

where x is the n-dimensional input vector, μ is an n-dimensional vector called centre of the RBF, $\| \cdot \|$ is the Euclidean distance, and ϕ is the RBF outline.

We built our model as a linear combination of N RBF functions with N centres. The RBF output is given by (6):

$$\hat{y}(x) = \sum_{j=1}^{N} \beta_j z_j(x), \tag{6}$$

where B_j is the weight of the jth radial function, centered at μ and with an activation level z_j.

RBF has seven inputs: two per each of the three motors in the robot arm (joints) and one for the label of the trajectory which is executed (T_r). Three inputs encode the difference between the current joint position and the next

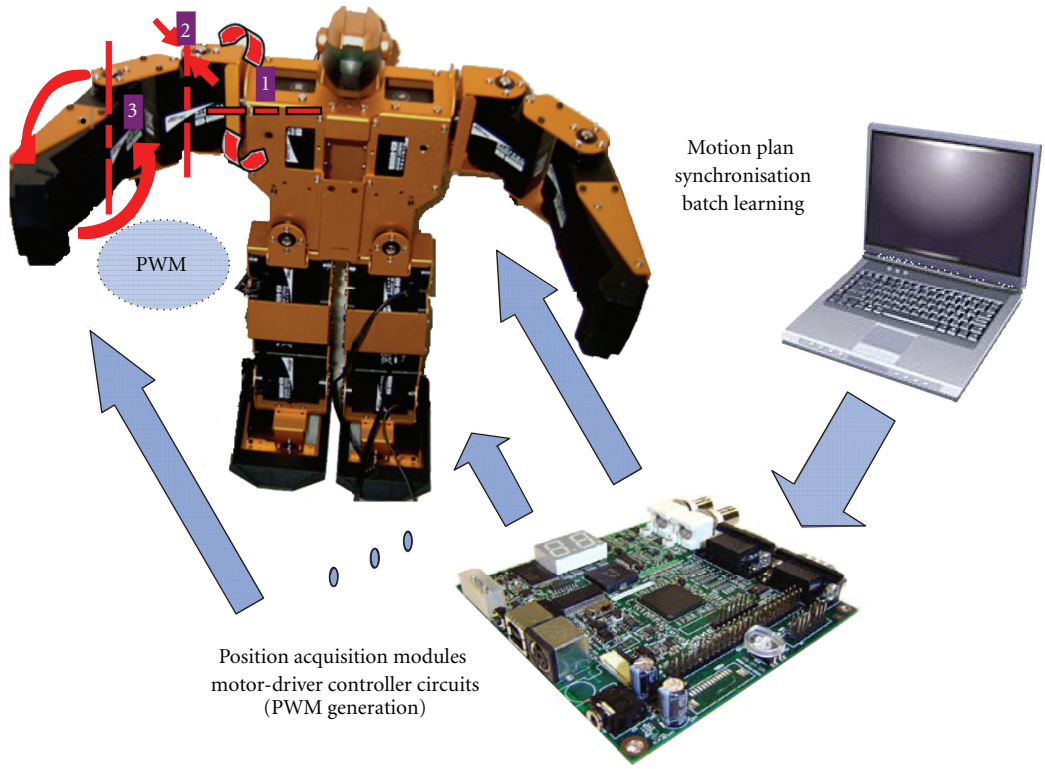

Figure 2: Diagram of the whole system architecture. The robot's picture in the left is showing the different joints of the 3 DOF Robonova arm involved in the movement. The PC applied appropriate motor commands to each joint of the arm at all times during the movement to follow the desired trajectory. To relieve the computer from interface computation and allow real-time communication with the robot, an FPGA board containing position acquisition modules and motor-driver controller circuits with PWM (pulse with modulation) was connected to the robot. The communication task between devices and the operations described above was synchronised by the computer speeding up the process. Finally, the batch learning with RBF of sensorimotor data collected during the movements was performed in the PC. The dataset was split up into training data (75%) and test data (25%).

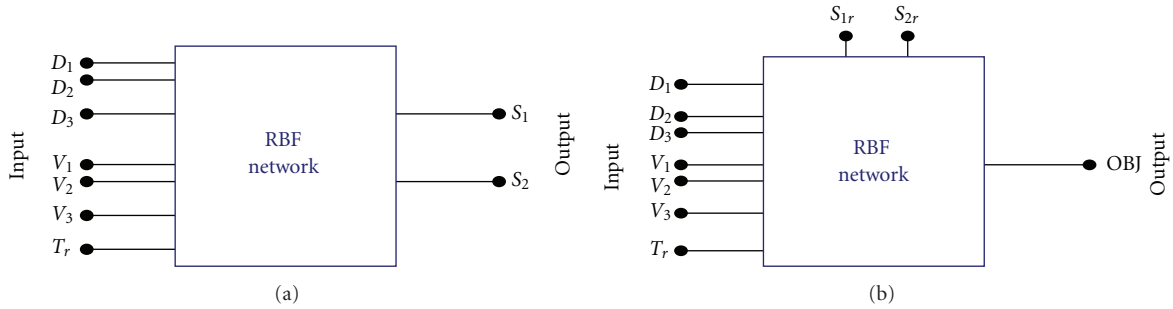

Figure 3: RBF inputs and outputs. (a) The network is used for function approximation of the sensor responses to specific inputs along different trajectories. (b) This network includes an output label (OBJ) for object classification and uses as inputs the actual sensor responses (connected to the upper part of the block).

target position in each of the joints (D_1, D_2, and D_3). The other three inputs (V_1, V_2, and V_3) encode the velocity computed on the basis of the previous three inputs. We aim to model the acceleration sensory outputs (S_1 and S_2) (S_3 does not provide further information than S_2 for the movement trajectory defined) and we also include one output (label) to classify different kinds of objects (OBJ). See the illustrative block in Figure 3(b). Therefore, we use two RBF networks. The first one (see Figure 3(a)) aims to approximate the acceleration sensor responses to the input motor commands. The second one (see Figure 3(b)) aims to discriminate object models (classification task) using as inputs the motor commands and also the actual acceleration estimations given by the sensors [57]. During learning, we feed the inputs and actual outputs of the training set in the network [27, 31]. During the test stage, we evaluate the error obtained from the sensors [39] that indicates how accurate the acquired model is and, therefore, how stable, predictable, and repeatable is

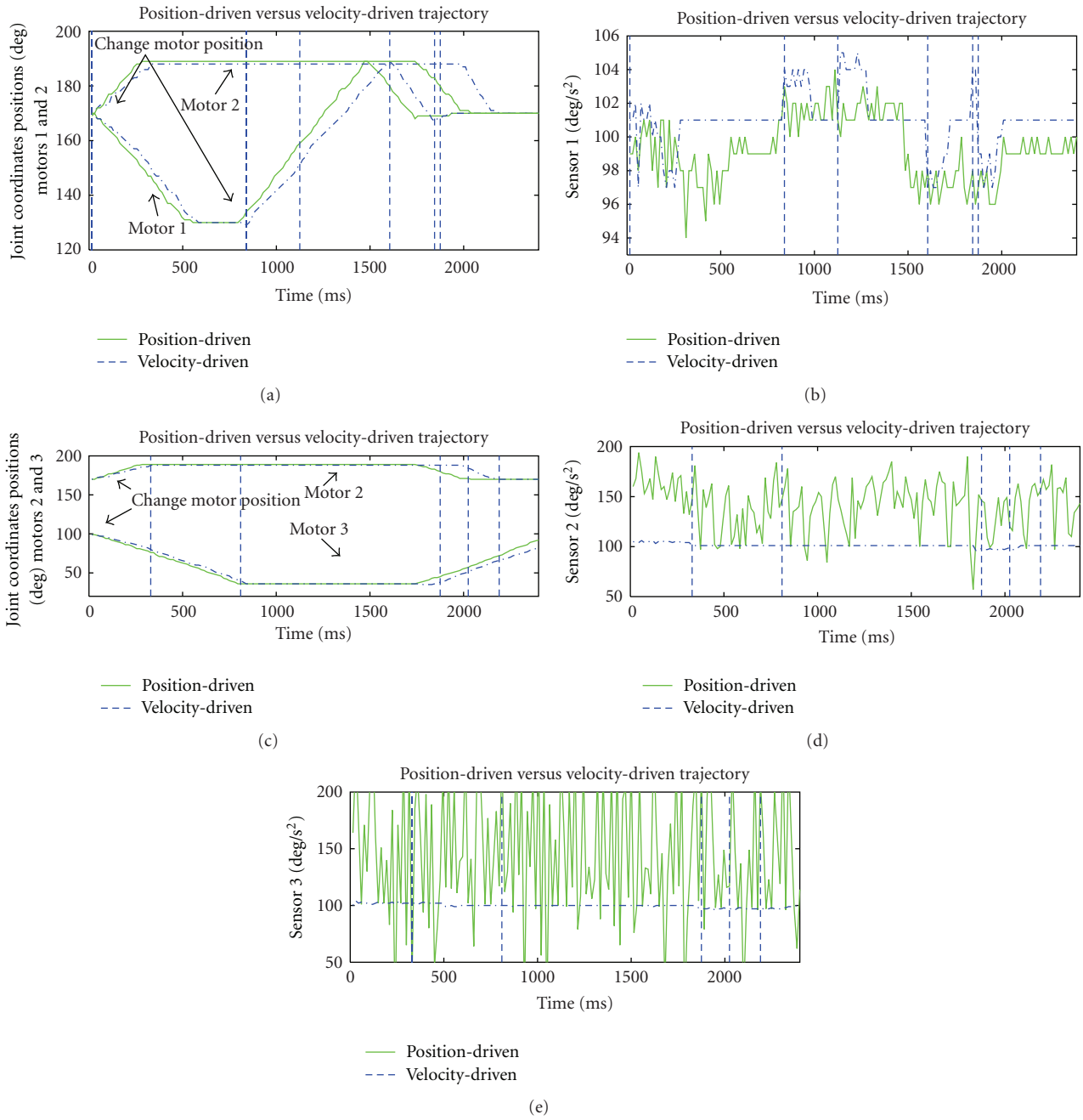

Figure 4: Plots (a) and (c) show the motor commands (in joint coordinates) and plots (b), (d), and (e) show the sensor responses during the movement. We compare the dense position-driven versus the sparse velocity-driven strategies for the case related to object 0 (see the experimental results). In (b) (Sensor 1), it can be clearly seen how the sensor responses are directly related with the motor command changes in plot (a). The sensor responses, obtained using the sparse velocity-driven strategy, are more stable. Motor commands increasing the derivative of joint coordinates cause increments in sensor 1 response during a certain period, while motor commands decreasing the derivative of joint coordinates cause decrements in sensor 1 response, plot (b). The piece of trajectory monitored by these plots is mainly related to sensor 1. It can be seen in the right plots (d) and (e) that the other sensors are not so sensitive to the motions in this example. Nevertheless, sparse velocity-driven control still leads to much more stable sensor responses. Dense position-driven control causes large vibrations that highly affect the capability of the neural network to approach the sensor response function.

the object manipulation. The error metric shown in the result figures (see Figure 5) is the root mean square error (RMSE) in degrees/s^2 of the different sensors along the whole trajectory. Finally, the classification error (related to the label output) indicates how accurately the network can classify a specific object from the positions, velocities, and accelerometer responses along the trajectory with a corresponding target output. A lower error in the predicted sensor responses indicates that the motion is performed in a smoother and more predictable way. Therefore, as can be seen in

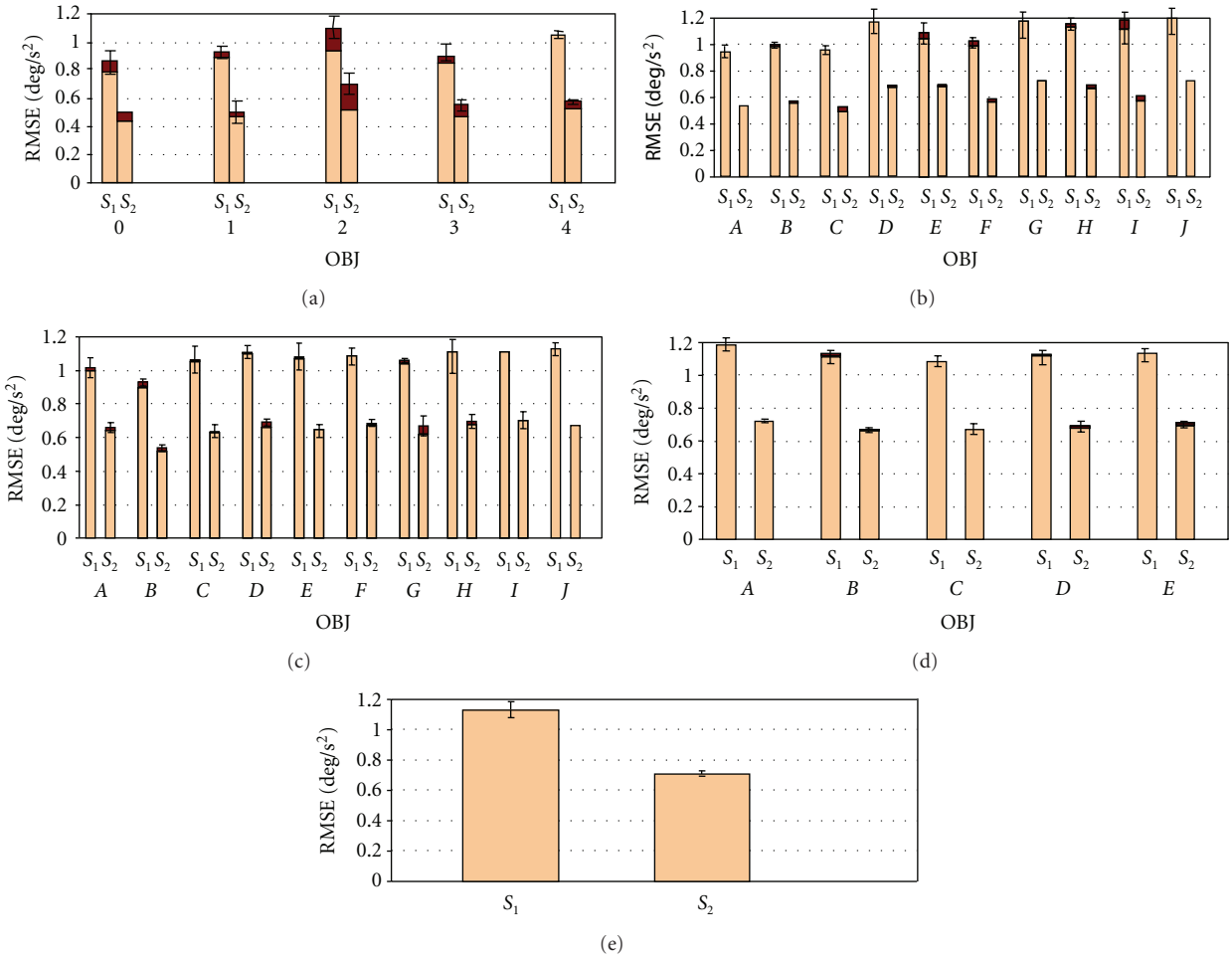

Figure 5: RMSE values for predicted data of sensors S_1 and S_2 for sparse velocity-driven control scheme. The darker colour in the top of the columns indicates the increment of RMSE in the test stage. (a) Using a different class (model) for each single object. (b) Using a different class (model) for each set of two objects. Cases (objects): *A* (0 and 1), *B* (0 and 2), *C* (0 and 3), *D* (0 and 4), *E* (1 and 2), *F* (1 and 3), *G* (1 and 4), *H* (2 and 3), *I* (2 and 4), and *J* (3 and 4). (c) Using a different class (model) for each set of three objects. Cases (objects): *A* (0, 1, and 2), *B* (0, 1, and 3), *C* (0, 1, and 4), *D* (0, 2, and 3), *E* (0, 2, and 4), *F* (0, 3, and 4), *G* (1, 2, and 3), *H* (objects 1, 2, and 4), *I* (1, 3, and 4), and *J* (2, 3, and 4). (d) Using a different class for each set of four objects. Cases (objects): *A* (0, 1, 2, and 4), *B* (0, 1, 2, and 3), *C* (0, 1, 3, and 4), *D* (0, 2, 3, and 4), *E* (1, 2, 3, and 4). (e) Using a single class for the only set of five objects. In this case the increment of RMSE in the test stage is neglectable. Cases (objects): (0, 1, 2, 3, and 4).

Figures 4, 5, and 6, strategies such as sparse velocity-driven control lead to a more reliable movement control scheme. Furthermore, this allows a better dynamic characterisation of different objects. This facilitates the use of adaptation strategies of control gains to achieve more accurate manipulation skills (although this issue is not specifically addressed in this work).

3. Results

Studies have shown high correlations between the inertia tensor and various haptic properties like length, height, orientation of the object, and position of the hand grasp [57–59]. The inertia tensor has the central role to be the perceptual basis for making haptic judgments of object properties.

In order to describe the approach and the pursued movements (several repetitions) and to test different dynamic models of the whole system (robot arm with object) [60, 61], the robot hand manipulated objects with different weights, shapes, and position of grasp. We acquired all the data from the position and accelerator sensors and we used them for the RBF off-line training. We considered the following cases of study:

(0) NO object;

(1) Scissors (0.05 kg, 0.14 m);

(2) Monkey wrench manipulated fixing the centre (0.07 kg, 0.16 m);

(3) Monkey wrench manipulated fixing an extreme (0.07 kg, 0.16 m);

(4) Two Allen keys (0.1 kg, 0.10 m).

We repeated manipulation experiments ten times for each case of study collecting sixty thousand data. We applied

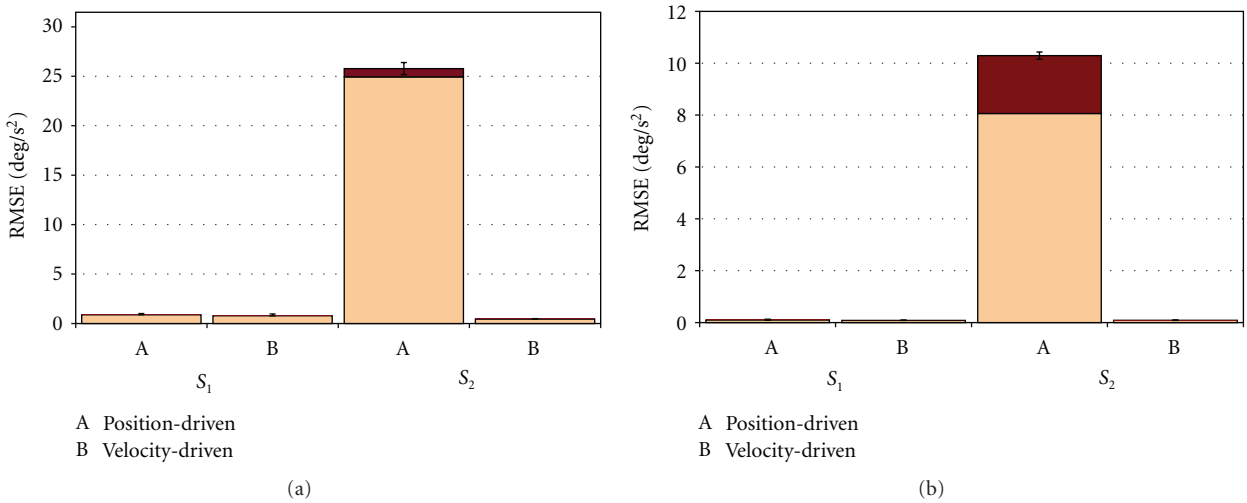

Figure 6: Comparison between dense position-driven control scheme (column A) and the sparse velocity-driven control scheme (column B) for the case related to object 0. RMSE (degrees/s^2) of predicted values of the sensors S_1 and S_2 using a sparse velocity-driven scheme (B) are always lower than the ones obtained using dense position-driven control (A). The sensor that suffers more significantly from jerks when using dense position-driven scheme is S_2. The darker colour in the top of the columns indicates the increment of RMSE in the test stage. The test results are always slightly higher. The variance value obtained with the cross-validation method for the test results is also included. Figures 6(a) and 6(b) are related to the cases of a nonfixed and a fixed robot respectively.

a cross-validation method defining different datasets for the training and test stages. We used the data of seven experiments out of ten for training and three for the test stage. We repeated this approach three times shuffling the experiments of the datasets for training and test. The neural network is built using the MBC model toolbox of MATLAB [62]. We chose the following algorithms: **TrialWidths** [63] for the width selection and **StepItRols** for the Lambda and centres selection [64]. We measured the performance calculating the RMSE in degrees/s^2 in the acceleration sensory outputs (S_1 and S_2).

We addressed a comparative study evaluating how accurately the neural network can abstract the dynamic model using a dense position-driven and a sparse velocity-driven schemes. Plots in Figure 4 illustrate how the motor commands are highly related with specific sensor responses (mainly in sensor 1 along this piece of trajectory). Figures 4(a) and 4(c) illustrate a piece of movement. Figures 4(b), 4(d) and 4(e) show that sparse velocity-driven control leads to much more stable sensor responses.

3.1. Abstracting the Dynamic Model during Manipulation. The goal of the RBF network is to model the sensory outputs given the motor commands along well defined trajectories (manipulating several objects). The purpose is to study how accurate models can be acquired if the neural network is trained with individual objects or with groups of "sets of objects" (as a single class). When we select the target "models" to abstract in the training stage with the neural network, we can define several target cases: 5 classes for abstracting single object models (Figure 5(a)), 10 classes for abstracting models for pairs of objects (Figure 5(b)), 10 classes for abstracting models of sets of "three objects" (Figure 5(c)), 5 classes for models of sets of "four objects" (Figure 5(d)), and finally, a single global model for the set of five objects (Figure 5(e)). The neural network, trained with data of two objects, presents the minimum of RMSE using 25 neurons in the hidden layer. For three objects, we used 66–80 neurons depending on the specific case, and for four objects, 80 neurons. But even with these larger numbers of neurons, the network was able to accurately classify the objects in the case of sets of two of them. During manipulation, if the model prediction significantly deviates from the actual sensory outputs, the system can assume that the model being applied is incorrect and should proceed to "switch" to another model or to "learn modifications" on the current model. In this task, we evaluate the performance of the "abstraction process" calculating the RMSE between the model sensor response and the actual sensor response along the movement. Figure 6 compares the performance achieved between the two control schemes under study for the cases of a nonfixed (Figure 6(a)) and a fixed robot (Figure 6(b)). Results indicate that the velocity driven strategy reduces the error to very low values in both cases.

Results in Figure 5 are obtained using a sparse velocity-driven control scheme. In Figure 5(a), they show how the RBF achieves different performance when trying to abstract models of different objects. For objects 1 and 3, both sensors lead to similar RMSE values as their dynamic models are similar. Objects 2 and 4 lead to higher RMSE values since they have more inertia than others. As we can see (comparing the results of Figures 5(a) and 5(b), if the network has to learn the dynamic model of more objects in the same class (shared model learning) at the same time, the error rate increases slightly.

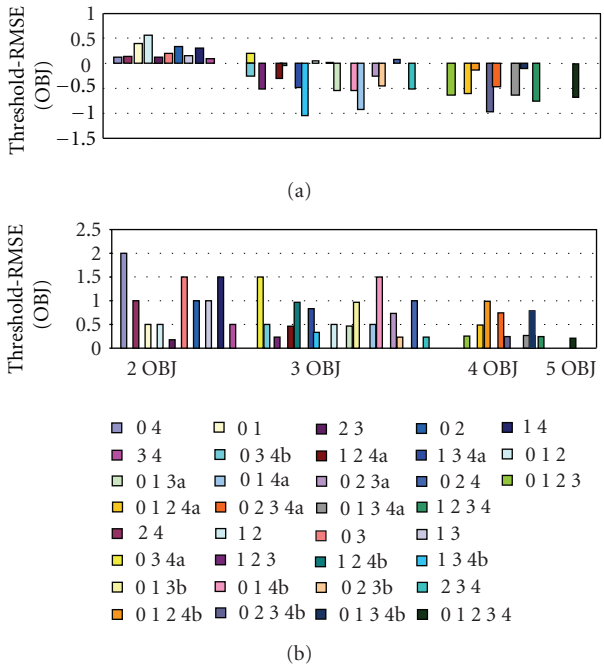

Figure 7: Bars represent the object discrimination power of the network (RBF) for all combinations of two, three, four, or five objects respectively (see paragraph 3 for object classification). If the RMSE (OBJ) value of the label output for a specific set of objects lies below the discrimination threshold, an accurate classification of objects can be achieved. Graphs show the result obtained with a non-fixed robot (Figure 7(a)) and with a fixed robot (Figure 7(b)), and each bar is associated to a combination of objects listed in the legend from left to right, respectively. A combination with two thresholds (i.e., 0 3 4) has two bars and two labels (0 3 4a and 0 3 4b) and bars are joined together in the graph.

3.2. Classifying Objects during Manipulation. In Figure 7, the discrimination power of the RBF based on the dynamic model itself is shown. In this case, we evaluate if the RBF is able to accurately distinguish among different objects assigning the correct labels (0, 1, 2, 3, and 4) by using only the inputs of the network (i.e., sensorimotor vectors). Evaluation of the classification performance is not straightforward and is based on the difference between the "discrimination threshold" (half of the distance between the closest target label values) and the RMSE value of the label output for a specific set of objects. We have chosen a single dimension distribution of the classes, therefore, distances between the different class labels are diverse. For instance, the "distance" between labels one and two is one, while the distance between labels one and four is three and the threshold is the half of that distance. For any set of two or five objects (i.e. A (0 and 1), B (1 and 3), and C (0, 1, 2, 3, and 4)) only one threshold value exists (Th(A) = 0.5, Th(B) = 1, and Th(C) = [0.5, 0.5, 0.5, 0.5]), while sets of three or four objects (i.e., D (1, 2, and 3), E (0, 2, and 3), and F (0, 1, 2, and 4), and G (1, 2, 3, and 4)) may have two threshold values (Th(D) = [0.5, 0.5], Th(E) = [1, 0.5], Th(F) = [0.5, 0.5, 1], and Th(G) = [0.5, 0.5, 0.5]). We see that the RBF is able to correctly classify the objects when using only two of them (values above zero)

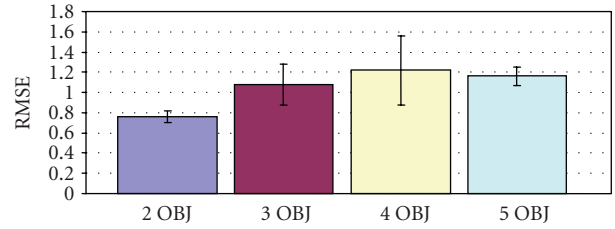

Figure 8: Mean of RMSE for objects (OBJ) using the sparse velocity-driven control scheme.

but the system becomes unreliable (negative values) when using three objects or more in the case of a standing robot (Figure 7(a)), while the RBF always correctly classifies the objects when the robot is fixed to the ground (Figure 7(b)).

3.3. Accurate Manipulation Strategy: Classification with Custom Control. We have abstracted dynamic models from objects and can use them for accurate control (Figure 9).

(a) Direct Control. We control different objects directly. During manipulation, the sensor responses are compared with a "shared model" to evaluate the performance. We measure the RMSE comparing the global model output (model of the sensor outputs of two different manipulated objects) and the actual sensor outputs.

(b) Classification with Control. Since we are able to accurately distinguish between two objects (Figure 7(a)), we select the corresponding individual model from the RBF network (see Figure 3(a)) to obtain the acceleration values of the object being manipulated instead of using a shared model with the data of the two objects. In fact, the individual models of the five objects were previously learnt. The RMSE is always lower when single object models are used (Figure 9) (after an object classification stage). Direct control using a global (shared) model achieves a lower performance because the applied model does not take into account the singularities of each of the individual models.

The *classification with control scheme* is feasible when dealing with objects easy to be discriminated and the actual trajectory of the movement follows more reliably the single object model than a "shared model." In Figure 9, the RMSE is plotted when *comparing sensor model response* and the *actual sensor responses*. Figure 8 shows the accuracy of the classification task when training the network with different numbers of objects (classification errors along the trajectory). Nevertheless, we have shown (Figure 7) that with the objects used in our study, the network can only reliably classify the object (along the whole trajectory) when we focus on distinguishing between two objects.

4. Conclusions

The first conclusion of the presented work is that the applied control scheme is critical to facilitate reliable dynamic model abstraction. In fact, the dynamic models or computed

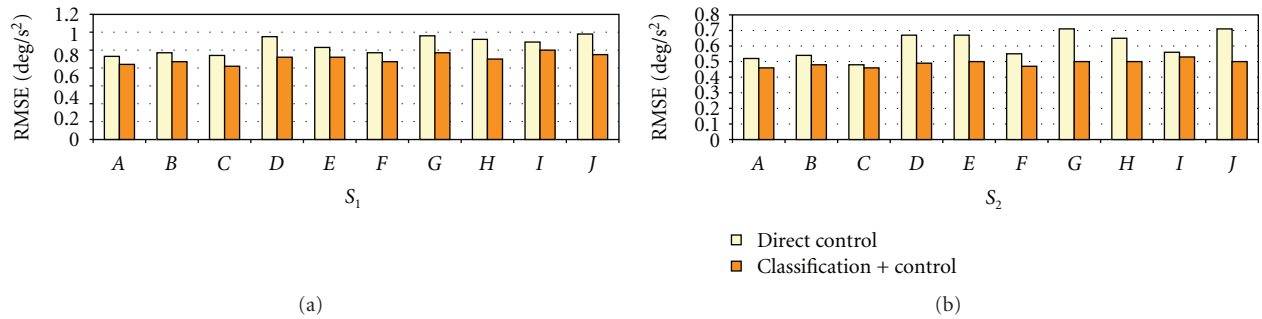

Figure 9: Comparison between the individual and the shared models related to predicted sensory values $S1$ and $S2$. The columns of a lighter colour indicate RMSE for the direct control while the columns of a darker colour represent RMSE for the controller of a single object once it is accurately classified. Cases (objects): A (0 and 1), B (0 and 2), C (0 and 3), D (0 and 4), E (1 and 2), F (1 and 3), G (1 and 4), H (2 and 3), I (2 and 4), and J (3 and 4).

torques are used for high performance and compliant robot control in terms of precision and energy efficiency [3, 65].

We have compared a dense position-driven versus a sparse velocity-driven control strategy. The second scheme leads to smoother movements when manipulating different objects. This produces more stable sensor responses which allow to model the dynamics of the movement. As model abstraction engine, we have used a well-known RBF network (widely used for function approximation tasks) and we compared its performance for abstracting dynamics models of the different objects using a sparse velocity-driven scheme (Figure 5(a)).

In a second part of the work, we have evaluated if the object classification is feasible. More concretely, we have checked out if we can perform it using sensorimotor complexes (motor commands, velocities, and local accelerations) obtained during object manipulation (exploration task) as inputs. Figure 7(a) shows that only the discrimination between any pair of two objects was possible. Robots with low gains will produce different actual movements when manipulating different objects, because these objects affect significantly their dynamic model. If the robot is controlled with high gains, the differences in the dynamic model of the robot are compensated by the high control gains. If the robot is tightly fixed but is controlled with low gains the actual movement (sensorimotor representation of an object manipulation) is different for each object (Figure 7(b)). If the robot is slightly fixed, the control commands produce vibrations that perturb the dynamic model and, therefore, the different sensorimotor representations are highly affected by these noisy artefacts (motor command driven vibrations) (see Figure 7(a)).

Finally, by assuming that we are able to classify objects during manipulation and to switch to the best dynamic model, we have evaluated the accuracy when comparing the movement's actual dynamics (actual sensor responses) with the abstracted ones. Furthermore, we have evaluated the gain in performance if we compare it with a specific dynamic model instead of a "global" dynamics model. We have evaluated the impact of this two-step control strategy obtaining an improvement of 30% (Figure 9) in predicting the sensor outputs along the trajectory. This abstracted model can be used in predicted control strategies or for stabilisation purposes.

Summarising, our results prove that the presented robot and artificial neural network can abstract dynamic models of objects within the stream sensorimotor primitives during manipulation tasks. One of the most important results of the work shown in Figures 6(a) (nonfixed robot) and 6(b) (fixed robot) indicates that robot platforms can highly benefit from nonrigid sparse velocity-driven control scheme. The high error obtained with the dense position-driven control scheme (Figure 6(a)) is caused by vibrations as the slight fixation of the robot to the ground is only supported by its own weight. Furthermore, when we try to abstract a movement model, the performance achieved by the neural network (RBF) also represents a measurement of the stability of the movements (the repeatability of such trajectories when applying different control schemes).

Acknowledgments

This work has been supported in part by the EU Grant SENSOPAC (FP6-IST-028056) and by the Spanish National Grants DPI-2004-07032 and TEC2010-15396.

References

[1] H. Hoffmann, S. Schaal, and S. Vijayakumar, "Local dimensionality reduction for non-parametric regression," *Neural Processing Letters*, vol. 29, no. 2, pp. 109–131, 2009.

[2] D. Nguyen-Tuong and J. Peters, "Learning robot dynamics for computed torque control using local Gaussian processes regression," in *Proceedings of the ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems (LAB-RS '08)*, pp. 59–64, Edinburgh, UK, August 2008.

[3] G. Petkos, M. Toussaint, and S. Vijayakumar, "Learning multiple models of nonlinear dynamics for control under varying contexts," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN '06)*, pp. 898–907, Athens, Greece, 2006.

[4] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[5] S. Vijayakumar, A. D'souza, T. Shibata, J. Conradt, and S. Schaal, "Statistical learning for humanoid robots," *Autonomous Robots*, vol. 12, no. 1, pp. 55–69, 2002.

[6] R. Miall, "Motor control, biological and theoretical," in *Handbook of Brain Theory and Neural Network*, M. Arbib, Ed., pp. 686–689, Bradford Books/MIT Press, Cambridge, Mass, USA, 2nd edition, 2003.

[7] S. Schaal and N. Schweighofer, "Computational motor control in humans and robots," *Current Opinion in Neurobiology*, vol. 15, no. 6, pp. 675–682, 2005.

[8] Sensopac, Eu project(SENSOrimotor structuring of Perception and Action for emergent Cognition), 2010, http://www.sensopac.org/.

[9] C. Q. Huang, S. J. Shi, X. G. Wang, and W. K. Chung, "Parallel force/position controllers for robot manipulators with uncertain kinematics," *International Journal of Robotics and Automation*, vol. 20, no. 3, pp. 158–167, 2005.

[10] J. Roy and L. L. Whitcomb, "Adaptive force control of position/velocity controlled robots: theory and experiment," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 121–137, 2002.

[11] M. Kawato, "Internal models for motor control and trajectory planning," *Current Opinion in Neurobiology*, vol. 9, no. 6, pp. 718–727, 1999.

[12] D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, vol. 11, no. 7-8, pp. 1317–1329, 1998.

[13] D. H. Shin and A. Ollero, "Mobile robot path planning for fine-grained and smooth path specifications," *Journal of Robotic Systems*, vol. 12, no. 7, pp. 491–503, 1995.

[14] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 103, no. 2, pp. 126–133, 1981.

[15] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, John Wiley & Sons, New York, NY, USA, 2006.

[16] W. Yu and M. A. Moreno-Armendariz, "Robust visual servoing of robot manipulators with neuro compensation," *Journal of the Franklin Institute*, vol. 342, no. 7, pp. 824–838, 2005.

[17] W. Yu and X. Li, "PD control of robot with velocity estimation and uncertainties compensation," *International Journal of Robotics and Automation*, vol. 21, no. 1, pp. 1–9, 2006.

[18] J. J. de Rubio and L. A. Soriano, "An asymptotic stable proportional derivative control with sliding mode gravity compensation and with a high gain observer for robotic arms," *International Journal of Innovative Computing, Information and Control*, vol. 6, no. 10, pp. 4513–4525, 2010.

[19] J. de Jesus Rubio, C. Torres, and C. Aguilar, "Optimal control based in a mathematical model applied to robotic arms," *International Journal of Innovative Computing, Information and Control*, vol. 7, no. 8, pp. 5045–5062, 2011.

[20] M. Haruno, D. M. Wolpert, and M. Kawato, "MOSAIC model for sensorimotor learning and control," *Neural Computation*, vol. 13, no. 10, pp. 2201–2220, 2001.

[21] B. Daya, "Multilayer perceptrons model for the stability of a bipedal robot," *Neural Processing Letters*, vol. 9, no. 3, pp. 221–227, 1999.

[22] S. Tolu, E. Ros, and R. Agís, "Bio-inspired control model for object manipulation by humanoid robots," in *Proceedings of the Proceedings of the 9th international work conference on Artificial neural networks (IWANN '07)*, pp. 822–829, San Sebastián, Spain, June 2007.

[23] R. R. Carrillo, E. Ros, C. Boucheny, and O. J. M. D. Coenen, "A real-time spiking cerebellum model for learning robot control," *BioSystems*, vol. 94, no. 1-2, pp. 18–27, 2008.

[24] N. R. Luque, J. A. Garrido, R. R. Carrillo, O. J.-M. D. Coenen, and E. Ros, "Cerebellar input configuration toward object model abstraction in manipulation tasks," *IEEE Transactions on Neural Networks*, vol. 22, no. 8, pp. 1321–1328, 2011.

[25] N. R. Luque, J. A. Garrido, R. R. Carrillo, O. J.-M. D. Coenen, and E. Ros, "Cerebellarlike corrective model inference engine for manipulation tasks," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 41, no. 5, pp. 1299–1312, 2011.

[26] 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," *International Journal of Neural Systems*, vol. 21, no. 5, pp. 385–401, 2011.

[27] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*, Massachusetts Institute of Technology, Cambridge, Mass, USA, 1995.

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

[29] J. Constantin, C. Nasr, and D. Hamad, "Control of a robot manipulator and pendubot system using artificial neural networks," *Robotica*, vol. 23, no. 6, pp. 781–784, 2005.

[30] T. D'Silva and R. Miikkulainen, "Learning dynamic obstacle avoidance for a robot arm using neuroevolution," *Neural Processing Letters*, vol. 30, no. 1, pp. 59–69, 2009.

[31] C. C. Lee, P. C. Chung, J. R. Tsai, and C. I. Chang, "Robust radial basis function neural networks," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 29, no. 6, pp. 674–685, 1999.

[32] C. Chez, *In Principles of Neural Science*, Prentice Hall, London, UK, 3rd edition, 1991.

[33] M. Ito, "Mechanisms of motor learning in the cerebellum," *Brain Research*, vol. 886, no. 1-2, pp. 237–245, 2000.

[34] C. Assad, S. Trujillo, S. Dastoor, and L. Xu, "Cerebellar dynamic state estimation for a biomorphic robot arm," in *Proceedings of the International Conference on Systems, Man and Cybernetics*, pp. 877–882, Waikoloa, Hawaii, USA, October 2005.

[35] M. Ito, "Control of mental activities by internal models in the cerebellum," *Nature Reviews Neuroscience*, vol. 9, no. 4, pp. 304–313, 2008.

[36] P. Van Der Smagt, "Cerebellar control of robot arms," *Connection Science*, vol. 10, no. 3-4, pp. 301–320, 1998.

[37] R. C. Miall, J. G. Keating, M. Malkmus, and W. T. Thach, "Simple spike activity predicts occurrence of complex spikes in cerebellar Purkinje cells," *Nature neuroscience*, vol. 1, no. 1, pp. 13–15, 1998.

[38] R. Apps and M. Garwicz, "Anatomical and physiological foundations of cerebellar information processing," *Nature Reviews Neuroscience*, vol. 6, no. 4, pp. 297–311, 2005.

[39] Ikarus, "Acceleration sensors," 2011, http://www.ikarus-modellbau.de.

[40] V. J. Lumelsky, M. S. Shur, and S. Wagner, "Sensitive skin," *IEEE Sensors Journal*, vol. 1, no. 1, pp. 41–51, 2001.

[41] C. J. Khoh and K. K. Tan, "Adaptive robust control for servo manipulators," *Neural Computing and Applications*, vol. 12, no. 3-4, pp. 178–184, 2003.

[42] M. Zhihong, X. H. Yu, and H. R. Wu, "An RBF neural network-based adaptive control for SISO linearisable nonlinear systems," *Neural Computing and Applications*, vol. 7, no. 1, pp. 71–77, 1998.

[43] Celoxica Inc, 2011, http://www.celoxica.com/.

[44] Hitec Robotics Inc, 2011, http://www.robonova.de/store/home.php.

[45] C. G. Atkeson, C. H. An, and J. M. Hollerbach, "Estimation of inertial parameters of manipulator loads and links," *International Journal of Robotics Research*, vol. 5, no. 3, pp. 101–119, 1986.

[46] D. Kubus, T. Kröger, and F. M. Wahl, "On-line rigid object recognition and pose estimation based on inertial parameters," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 1402–1408, San Diego, Calif, USA, November 2007.

[47] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, Pearson/Prentice Hall, Upper Saddle River, NJ, USA, 3rd edition, 2005.

[48] M. Krabbes, C. Doschner et al., "Modelling of robot dynamics based on a multidimensional rbf-like neural network," in *Proceedings of the International Conference on Information Intelligence and Systems (ICIIS'99)*, pp. 180–187, Bethesda, Md, USA, October/November 1999.

[49] R. J. Shilling, *Fundamentals of Robotics: Analysis and Control*, Prentice Hall, Englewood Cliffs, NJ, USA, 1990.

[50] P. Angelov, "Fuzzily connected multimodel systems evolving autonomously from data streams," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 41, no. 4, pp. 898–910, 2011.

[51] P. Angelov and D. Filev, "Simpl_eTS: a simplified method for learning evolving Takagi-Sugeno fuzzy models," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 1068–1073, Reno, Nev, USA, May 2005.

[52] G. P. Liu, V. Kadirkamanathan, and S. A. Billings, "Variable neural networks for adaptive control of nonlinear systems," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 29, no. 1, pp. 34–43, 1999.

[53] H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran, "Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction," *Fuzzy Sets and Systems*, vol. 157, no. 9, pp. 1260–1275, 2006.

[54] J. de Jesús Rubio, D. M. Vázquez, and J. Pacheco, "Backpropagation to train an evolving radial basis function neural network," *Evolving Systems*, vol. 1, no. 3, pp. 173–180, 2010.

[55] J. J. Rubio, F. Ortiz-Rodriguez, C. Mariaca-Gaspar, and J. Tovar, "A method for online pattern recognition of abnormal-eye movements," *Neural Computing & Applications*. In press.

[56] D. K. Wedding and A. Eltimsahy, "Flexible link control using multiple forward paths, multiple RBF neural networks in a direct control application," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 2619–2624, Nashville, Tenn, USA, October 2000.

[57] Z. D. Wang, E. Nakano, and T. Takahashi, "Solving function distribution and behavior design problem for cooperative object handling by multiple mobile robots," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 33, no. 5, pp. 537–549, 2003.

[58] J. Flanagan, K. Merritt, and R. Johansson, "Predictive mechanisms and object representations used in object manipulation," in *Sensorimotor Control of Grasping: Physiology and Pathophysiology*, pp. 161–177, Cambridge University Press, Cambridge, UK, 2009.

[59] C. C. Pagano, J. M. Kinsella-Shaw, P. E. Cassidy, and M. T. Turvey, "Role of the inertia tensor in haptically perceiving where an object is grasped," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 20, no. 2, pp. 276–285, 1994.

[60] J. S. Bay, "Fully autonomous active sensor-based exploration concept for shape-sensing robots," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, no. 4, pp. 850–860, 1991.

[61] G. Petkos and S. Vijayakumar, "Load estimation and control using learned dynamics models," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 1527–1532, San Diego, Calif, USA, November 2007.

[62] Mathworks Inc, 2011, http://www.mathworks.com/products/matlab.

[63] M. Orr, J. Hallam, K. Takezawa et al., "Combining regression trees and radial basis function networks," *International Journal of Neural Systems*, vol. 10, no. 6, pp. 453–465, 2000.

[64] S. Chen, E. S. Chng, and K. Alkadhimi, "Regularized orthogonal least squares algorithm for constructing radial basis function networks," *International Journal of Control*, vol. 64, no. 5, pp. 829–837, 1996.

[65] D. Nguyeii-Tuoiig, M. Seeger, and J. Peters, "Computed torque control with nonparametric regression models," in *Proceedings of the American Control Conference (ACC '08)*, pp. 212–217, Seattle, Wash, USA, June 2008.