On-line Learning and Control of Attraction Basins for the Development of Sensorimotor Control Strategies

Antoine de Rengervé, Pierre Andry, Philippe Gaussier

Biological Cybernetics manuscript No.
(will be inserted by the editor)

On-line Learning and Control of Attraction Basins for the Development of Sensorimotor Control Strategies

Antoine de Rengervé · **Pierre Andry** · **Philippe Gaussier**

Received: date / Accepted: date

Abstract Imitation and learning from human require an adequate sensorimotor controller to learn and to encode the behaviors. We present the Dynamic Muscle PerAc (DM-PerAc) model to control a multi DOF robot arm. In the original Perception-Action (PerAc) model, path following or place reaching behaviors correspond to the sensorimotor attractors resulting from the dynamics of learned sensorimotor associations. The DM-PerAc model, inspired by human muscles, permits to combine impedance-like control with the capability of learning sensorimotor attraction basins. We detail a solution to incrementally learn on-line the DM-PerAc visuomotor controller. Postural attractors are learned by adapting the muscle activations in the model depending on movement errors. Visuomotor categories merging visual and proprioceptive signals are associated with these muscle activations. Thus, the visual and proprioceptive signals activate the motor action generating an attractor which satisfies both visual and proprioceptive constraints. This visuomotor controller can serve as a basis for imitative behaviors. Besides, the muscle activation patterns can define directions of movement instead of postural attractors. Such patterns can be used in state-action couples to generate trajectories like in the PerAc model. We discuss a possible extension of the DM-PerAc controller by adapting the Fukuyori's controller based on the Langevin's equation. This controller can serve not only to reach attractors which were not explicitly learned but also to learn the state/action couples to define trajectories.

Keywords visuomotor control · impedance control · perception-action loop · neural network

ETIS UMR CNRS 8051, ENSEA, University Cergy Pontoise F-95000 Cergy Pontoise, France
E-mail: {rengerve,andry,gaussier}@ensea.fr),

1 Introduction

In order to act efficiently in unknown environment and collaborate with human, robots must be able to control and adapt their behaviors. On the contrary of classical motor control approach, Human-Robot Interaction and imitation paradigms take into account that a human partner can influence and improve both the behavior and the behavioral learning of a robot. Our past work, following a developmental approach [47], along with collaborations with developmental psychologists, cognitive psychologists and neurobiologists have led us to understand that the tasks and behaviors cannot be reduced to a set of controlled parameters. Behaviors rather emerge from the dynamics of perception action coupling [26, 48]. The behavior is built upon a wide range of interactions at different levels. A behavior learning system must be able to capture the dynamical *sensorimotor attractors* describing the behaviors. In such conditions, the issues of learning, adapting and sharing these attractors are fundamental in order to achieve natural and intuitive non verbal human-robot interaction. What are the constraints on the low level motor control to learn such attractors? What kind of model of motor control should be used and how can it be learned ?

Impedance control enhances optimal control in the case of interaction with the environment (Sec. 2.1). In impedance control, position and velocity constraints determine the movements with respect to the desired trajectory. In the framework of human robot interaction, regression based solutions [37, 10] can learn the desired trajectories from data obtained during the task demonstration by a human (Sec. 2.2). The trajectories result from mixtures of adapted kernels. Impedance control can be linked to muscle activations (Sec. 2.3). Though, the hypothesis of a desired trajectory is usually kept while focusing on the link between muscle activations and the impedance control parameters (stiffness,…). On the contrary, we defend the Perception-Action (PerAc) approach claiming that behaviors correspond to sensorimotor attractors emerging from the dynamics of multiple learned sensorimotor associations (Sec. 3).
In our first works on the emergence of imitation [27, 3], we showed that an arm controller using the learning of visuomotor associations to build an homeostatic controller can lead to the emergence of low level imitative behaviors if the perception is ambiguous (i.e. when mistaking partner's hand for its own hand). However, this visuomotor controller had several limitations. In particular, it did not allow the coding of trajectories by state-action couples like in the PerAc approach. We thus propose, in this paper, a model called Dynamic-Muscle PerAc to control a robot arm with multiple Degrees-of-Freedom (Sec. 4). The DM-PerAc model is based on simple models of muscles and joints with dynamic equations corresponding to impedance control. This

DM-PerAc model learns the inverse kinematic model by learning visuomotor associations. It also learns postural attractors to link perception (visuomotor categories) with actions coded as muscle activations i.e. it also learns the inverse dynamic model. The behavior and properties of the DM-PerAc visuomotor controller are evaluated in Section 5. Like in our previous works [3], the DM-PerAc visuomotor controller is a good bootstrap for imitative behaviors (Sec. 6.2). Besides, the muscle activation patterns can be used in state/action couples to code trajectories like in the PerAc model (Sec. 6.1). We introduce the Fukuyori's controller to improve performance in Section 6.3 and we discuss its possible role to learn trajectories with the DM-PerAc model in Section 7.

2 State of the art of on-line, incremental motor control for learning from interaction

2.1 Impedance control

In optimal control theory [62], the desired trajectory is an optimal trajectory crossing given via-points and minimizing some movement variables like jerk[1] [21]. The motor control should be flexible enough to allow physical interaction with the environment. Studies of movement properties have led to impedance control model [35] as an approximation of neuro-muscular properties. According to the equilibrium trajectory hypothesis [20], motor programs are internally represented as the trajectories of an equilibrium point. Impedance control is efficient to control manipulators acting in contact with the world [15]. Impedance control is also a usual controller for prostheses and exoskeleton which involve direct physical interaction with a human [41]. Impedance control is based on a second order "damped mass spring"-like system (1) enabling constrained motion, dynamic interaction and obstacle avoidance.

$$M\frac{dV}{dt} = K(X_0 - X) + B(V_0 - V) \tag{1}$$

with V the velocity and X is the Cartesian position of the end effector. The coefficient K (equivalent to the spring stiffness) and B represent the constraints related to the position command X_0 and the speed command V_0 respectively. Some other versions of impedance control use the proprioceptive information (e.g. [1]) instead of the Cartesian position. Besides, the via-points, which are necessary to compute the desired trajectory $(X_0(t), V_0(t))$, can be learned from watching [49].

[1] In the minimum-jerk approach, the movements maximize the smoothness of the motion.

2.2 Learning tasks from human with regression techniques

The trajectories can be directly learned from training data obtained during a task demonstration by a human. In order to learn how to fulfill a task, a human teacher can provide feedback or data which are integrated in a sensorimotor model of the task. Function approximation based on local regression techniques [5] is efficient to learn forward or inverse models of robot control. Learning an initial model from a human demonstration reduces the size of the space to be explored. Demonstrations facilitate and improve a subsequent reinforcement learning [57]. More recent, the Locally Weighted Projection Regression algorithm (LWPR) [63] merges both the incremental learning properties of the Receptive Field Weighted Regression (RFWR) algorithm [59] and the projection of input data in order to reduce the dimensionality problem. The authors showed a demonstration with a 30-DOF SARCOS humanoid robot learning the dynamic inverse model and performing eight-shaped trajectories with its arm.
Regression techniques to learn models of motor control were also used in learning from demonstration paradigm [4]. The Dynamic Movement Primitives (DMP) [37, 58, 34, 38] are based on the RFWR algorithm. The primitives are control policies that are activated depending on a local basis function. They provide motor control as a second order dynamic system. The combination of primitives shapes the attractor landscape to produce the desired trajectory. This combination depends on a phase variable which gives the temporal reference of the movement. The approximated function is the time-dependent trajectory, and locally weighted regression of training data determines the parameters of the basis functions (number, centers, bandwidths) and the contribution of corresponding primitives. The DMP algorithm shows interesting properties of spatial and temporal invariance and was applied to learn discrete and rhythmic movements. However, the correspondence problem [50] was completely eluded as the training data were obtained from joint-angle recording system on the human. A particular coupling must be introduced in the dynamic equation of the phase variable in order to tackle correctly perturbations. The action of this coupling is to slow the evolution of the phase variable when there are perturbations.
Similarly, a Gaussian Mixture Model (GMM) can also learn a model of a demonstrated task by encoding proprioceptive and Cartesian information in Gaussian kernels [10]. The learning is based on an Expectation-Maximization process which adapts the Gaussian kernels to describe probabilistically the input data obtained in a training session. Then, given partial information like only the Cartesian position, Gaussian Mixture Regression extracts the probable proprioception to control a robotic arm. Depending on the task, vision or motion capture devices can track particular ele-

ments (e.g. spoon, human head) [12, 13]. Still, the computation of the 3D Cartesian coordinates of the visual markers requires particular calibrations of the external devices. [11] uses a dynamical second order motor controller and Hidden Markhov Models (HMM) instead of GMM. HMM encodes the sequential dependencies in the task, whereas the motor controller now implements impedance control. A trade-off between the position constraint and the speed constraint is managed depending on the variance in the demonstrated trajectories. This version of the model is similar to DMP. The main difference is that the learning of the constraints on the position and the velocity profile can take into account the mutual influence between different Degrees-of-Freedom which is not the case with DMP. Some recent works [44, 54] studied the on line adaptation of the control stiffness from the position variations and haptic feedback. This adaptation of the control improved the quality of the collaboration between Human and robot [54].

2.3 Adaptation of muscle activations and impedance control

In the case of human arm control, the actions are generated by muscle contraction. The VITE model [7] is based on equations describing the muscle activations. The resulting dynamics is similar to the dynamics produced by an impedance controller [32]. However, the VITE model also assumes a target position to drive muscle activations. In iterative and adaptive control [61], the behavior can be adapted by changing the control parameters instead of changing the command. Considering the adaptation properties at the level of muscular control [8, 22], the authors proposed a muscle centered model of adaptive and iterative control to maintain a posture or to follow a trajectory under disturbances [25]. The controller takes into account a feedforward torque command and a feedback control to generate the final torque command. The feedforward torque command is generated by muscular activation. The feedback controller is proportional derivative. Such control can be equivalent to impedance control if the apparent inertia is assumed to vary and to be equal to the inherent inertia of the robot. The muscle activations are adapted in order to reduce the feedback error. Indeed, in the model [25], the adaptation of the muscle activities directly induces changes of the feedforward torque and of the stiffness in the feedback controller. Feedforward torque modification enables to compensate an applied external force. In the case of rapidly varying disturbances, the stiffness of the feedback controller is increased, so the robustness of the controller also increases. However, increasing the stiffness from a muscular point of view is energy consuming. So, the stiffness will tend to decrease when the unpredictable perturbations cease to occur. This model permitted to maintain a desired posture or to follow an a-priori given trajectory. The principle of adapting the muscle activations should not be reduced to adapting the parameters of the impedance control. This principle is also interesting to learn the perception-action coupling.

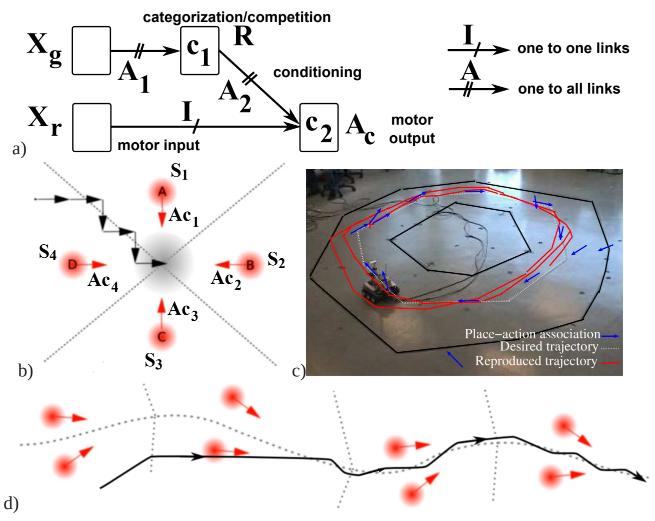

Fig. 1 *a)* PerAc model. *b-d)* Examples of built dynamics in 2D spaces. *b)* Fixed point attractor. *c)* Limit cycle in the case of navigation experiment. *d)* Trajectory following. In b) and d), the gray dotted lines are the Voronoi boundaries. The plain black line is a trajectory sample.

3 The Perception-Action model and arm control

Since many years, we have defended the Perception-Action approach (PerAc, [26]) claiming that, in an active system, coupling perception and action enables to build behaviors. Fast on-line learning of associations between sensory signals and motor signals is sufficient to build sensorimotor attraction basins. Let us consider the sensorimotor system of an agent acting in a given environment (or state space) and having 2 sensation vectors $\mathbf{X}_r$ and $\mathbf{X}_g$ (Fig. 1a). Firstly, the proprioception vector $\mathbf{X}_r$ represents the coarse feedback information from the execution of the motor command or the direction of the goal (if the goal is in immediate neighborhood). It can be considered as a reflex or a regulatory pathway that links proprioceptive sensation to the motor command **Ac**. Secondly, the global sensory vector $\mathbf{X}_g$ represents more global information about the environment. A local but robust distance measure (metric) can be computed to compare global sensory vectors. In the PerAc model (Fig. 1a), the global sensory vector is categorized and a competition (soft-WTA) between the categories allows to define recognition activities **R**. On the basis of the distance measure, the categories which best represent the current state are determined. Categories are associated with concurrent actions estimated from the proprioceptive vector $\mathbf{X}_r$. An action field is thus defined. This action field associates particular actions (movement vectors or forces) to areas of the state space according to the recognized categories. Depending on the built action field, the dynamics of the system can be shaped to produce interesting behaviors i.e. attractor points, limit cycles or trajectories. Figure 1b-d shows examples of dynamics defined in a 2D space. In Figure 1b and d, the Voronoi diagram shows for any point of the space which category wins the recognition competition. The associated action is thus performed as long as the state of the system is in the same Voronoi area. A trajectory sample is given in Figure 1b. The system reaches the boundary of the Voronoi area where it started, then it follows this boundary to the defined attractor point. Whatever the initial position is, the learned dynamics leads the system to the attractor point with a similar kind of trajectory. The attraction basin emerges from the system dynamics generated by the state/action couples. Figure 1c shows a configuration of action field that produces a limit cycle. No time basis is necessary. As the system moves, it reaches another area of the action field and performs the corresponding action which brings and maintains the system close to the followed limit cycle. Not using a time basis has several advantages. No synchronization of the time reference is needed, which is quite a complex process, especially when there are perturbations of the trajectory. The learning is also more direct, and can be performed on-line very rapidly because the model simply learns what should be done in a directly sensed context.

A similar kind of state/action combination can also produce a simple trajectory following (Fig. 1d) Indeed, partial limit cycle construction can provide a dynamics with which the system behaves as if it is “attracted” by a trajectory and remains in its close vicinity. In the state/action configuration of Fig. 1d, the system can only get closer to an “equilibrium” path where, due to the alternate category recognition, the effects of the associated actions tend to equilibrate. The system is maintained in the vicinity of this path. Depending on the orientation of the learned movement actions, the system will tend more to reach the trajectory or to move forward. By allowing the system to come back to the trajectory, the PerAc model can manage perturbations.

The PerAc model has been proved to be an efficient control for navigation and path following [31], with good robustness against perturbations like obstacle avoidance. In these works, the learned categories are place-cells based on visual recognition of the robot location (see [31] for details). The state/action associations are learned on-line from interaction with a teacher [30]. When the robot moves away from the desired trajectory, the human teacher changes its orientation to correct its behavior. This feedback is used to learn new place-cell/orientation couples to complete the sensorimotor control and to modify the robot behavior. This sensorimotor learning enables the robot to follow trajectories (limit cycles, Fig. 1c) and even to reach particular locations which become attractors for the dynamical system. In the PerAc approach, the perception is considered as the result of learning sensation/action associations allowing a globally consistent behavior while facing an object. For instance, by learning sensorimotor associations, a robot can learn how to return to a given object which can be interpreted as the fact that the robot “perceives” the object [48].

The same sensorimotor association principle can be a basis for the emergence of low level imitative behaviors [27]. In the case of arm control, we showed [3] that an imitation of directly observed gestures can appear as a side effect of a homeostatic visuomotor controller with perceptual ambiguity. During a first phase, the system learns associations between visual and motor signals building a visuomotor *homeostat*. Due to low visual capabilities, the robot is unable to discriminate its own hand from the hand of a teacher (ambiguity of perception). As the control architecture implements a homeostat, the system tends to maintain the equilibrium between visual and proprioceptive information. If a difference is perceived, then the system acts to come back to the equilibrium state. To do so, the robot moves its arm so that its proprioceptive configuration corresponds to the perceived visual stimuli according to its sensorimotor learning. As a result of these movements, the demonstrator’s gestures are imitated [3]. The correspondence problem [50] is avoided as the robot only imitates what is observed with its own capabilities.

In the model of [3, 45], the control was performed in the visual space. A forward kinematic model allowed to estimate the visual position of the robot hand. This position was then compared with the perceived visual position to generate movements (see [3] for details). A first drawback was that erratic estimations of the visual position of the robot hand produced an erratic control. Because the forward model learning was based on Self-Organizing Maps [43], false estimations could occur until learning convergence. So, the controller should not be used before the end of learning. The learning process was not incremental. Finally, the trajectories were not coded by sensorimotor couples like in the PerAc model. Indeed, the motor commands were extracted from the Dynamic Neural Fields [60] by using an ad hoc readout mechanism. This solution presented interesting properties (memory, bifurcation) (see Sec. 5.4), but was only able to define attractor positions. Moreover, we were not able to explain how the readout process could be learned or tuned. Here, we are interested in a model that can bootstrap imitative behaviors and can also code trajectories according to the PerAc approach. The model should also be incremental and able to managed multiple Degrees of Freedom.

In [40] [3], the authors developed arm controllers which work in spaces different from the motor space, reducing the number of dimensions. The difficulty is then to extract a motor command from the control in the lower dimension space. In the DM-PerAc model, we use the alternate solution consisting in performing the control in the proprioceptive space. The generation of the motor command is simplified whereas the difficulty is to learn sensorimotor attractors. The resulting motor controller should be able to learn either a particular movement or a postural attractor. In the next section, we describe the Dynamic-Muscle PerAc (DM-PerAc) model which provides a common coding basis for both aspects of the control. The DM-PerAc model is based on a simplified model of joints and muscles where both particular movements and postural attractors are coded as muscular activations. We also detail how visuomotor attractor are learned by the DM-PerAc model.

4 Dynamic-Muscle PerAc model

We now present our model called Dynamic Muscle PerAc to control a robotic arm. This model combines control equivalent to impedance control with the PerAc principle. The parameters and equations of the DM-PerAc model are all summarized in Appendix A.

4.1 Control of joint position with a simplified muscle model

Different models like Hill's model [33] and Huxley's model [36] have been developed describing different properties of the muscles. In the lumped-parameter nonlinear antagonistic muscle model [64, 65], the movements of a joint are produced by a couple of antagonist muscles. The muscles are simulated by Hill's muscle model. This model is based on three components: a contractile element, a series elastic element and a parallel elastic element. In [42], the two elastic elements are neglected to focus on the dominant contractile element. The contractile element can be approximated by a force generator in parallel with a damping element [24]. The force generator implements the force-length relation in muscles with the force that can be modulated by neural signals [65]. The damping element implements the force-velocity relation given by [33].

Our model, called Dynamic-Muscle PerAc (DM-PerAc), is also based on couples of antagonist muscles (hereafter noted $+$ and $-$) around the joints with each muscle approximated as a contractile element. However, unlike [42] and [65], we use a simplified linear model of contractile element which generates torque instead of force. In the DM-PerAc model, the torque generator is a spring with variable stiffness whereas the damping element is a simple viscous damper (Fig. 2). The varying stiffness is given by the muscle activations **A**. The joint positions are controlled with the equations [2-8]. As these equations are the same for each joint, the joint index j is not displayed. Besides, the time step (t) dependency is only indicated to disambiguate terms when different time steps are involved in the same equation. For each joint, the agonist and the antagonist muscles generate the apparent torques τ^+ and τ^- (2).

$$\begin{cases} \tau^+ = -A^+ \cdot \theta^+ - \sigma^+ \cdot \dot{\theta}^+ \\ \tau^- = -A^- \cdot \theta^- - \sigma^- \cdot \dot{\theta}^- \end{cases} \tag{2}$$

where A^+ (resp. A^-) is the muscle activation and σ^+ (resp. σ^-) is the damping[2] of the agonist (resp. antagonist) muscle. The angular values θ^+ and θ^- are measured respectively from the full flexion position θ_{max} and from the full extension position θ_{min} (3).

$$\theta^+ = \theta - \theta_{max} \;\; , \;\; \theta^- = \theta - \theta_{min} \;\text{ and }\; \theta \in [\theta_{min}, \theta_{max}] \tag{3}$$

with θ the angular position of the joint.

The dynamical equation of the system links the rotational acceleration $\ddot{\theta}$ and the moment of inertia I with the torques generated by the agonist and antagonist muscles given by (2) and the torque τ_e given by external forces.

$$\begin{aligned} I \cdot \ddot{\theta} &= \tau^+ + \tau^- + \tau_e \\ &= -A^+ \cdot \theta^+ - \sigma^+ \cdot \dot{\theta}^+ - A^- \cdot \theta^- - \sigma^- \cdot \dot{\theta}^- + \tau_e \end{aligned} \tag{4}$$

Equations (3) and (4) gives the equation (5) where $\sigma = \sigma^+ + \sigma^-$:

$$I \cdot \ddot{\theta} = A^+ \cdot (\theta_{max} - \theta) - A^- \cdot (\theta - \theta_{min}) - \sigma \cdot \dot{\theta} + \tau_e \tag{5}$$

In the absence of external torques/forces ($\tau_e = 0$), the system defines an attractor at the convergence point $\theta_{eq} = \frac{A^+ \cdot \theta_{max} + A^- \cdot \theta_{min}}{A^+ + A^-}$. To simplify this controller, the angular positions θ of the joint are normalized so that for each joint, they vary between 0 and 1.

$$\theta_{min} = 0 < \theta < \theta_{max} = 1 \;,\;\; \theta^+ = 1 - \theta \;\text{ and }\; \theta^- = \theta \tag{6}$$

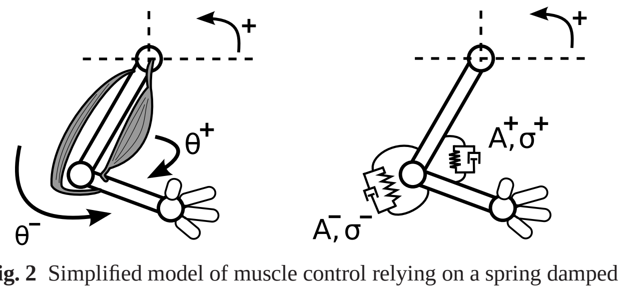

Fig. 2 Simplified model of muscle control relying on a spring damped model of muscles. Damping properties are hypothesized as mechanical property of the arm still related to the muscle stiffness.

[2] The damping can be constant. However, controlled movements are improved if the damping varies with the stiffness. For instance, the damping can be defined as proportional to the square root of the stiffness like in [25].

In this particular case, our control equation (5) is equivalent to (7) with $\theta_{eq} = \frac{A^+}{A^+ + A^-}$ with $K = A^+ + A^-$.

$$\ddot{\theta} = \frac{K}{I} \cdot (\theta_{eq} - \theta) - \frac{\sigma}{I} \cdot \dot{\theta} \tag{7}$$

The equation (7) corresponds to a classical mass-spring-damping system with a stiffness K and an equilibrium position θ_{eq}. The equilibrium position is unchanged when both A^+ and A^- are multiplied by the same factor. Such a factor only modifies the equivalent stiffness K. An adaptation of the stiffness K and the damping σ controls the rise time, overshoot and settling time. The controller was simulated using discrete time with a time increment Δt. With I the moment of inertia and τ the sum of the torques $\tau = \tau^+ + \tau^-$, the equations of the dynamical system are:

$$\begin{cases} \theta_t = \theta_{t-\Delta t} + \dot{\theta}_t \cdot \Delta t \\ \dot{\theta}_t = \dot{\theta}_{t-\Delta t} + \ddot{\theta}_t \cdot \Delta t \\ \ddot{\theta}_t = \tau_t / I \end{cases} \tag{8}$$

The variables $\theta_t, \dot{\theta}_t, \ddot{\theta}_t$ correspond respectively to $\theta, \dot{\theta}, \ddot{\theta}$ in the equations [2-7].

In our model (5), the generated torque depends on the activation **A** of the muscles and on the lengths of the muscles (indeed the angles θ). This dependance to the muscle length makes our model look like the ”lambda” model of Feldman [17, 18]. In the Theory of the Equilibrium Point [19], also named Theory of Threshold Control, the motor control is based on threshold functions (λ) defining the activation of the agonist and antagonist muscles. However, in our model, the activation thresholds are not controlled. The activation of the muscles is directly the controlled parameter. Therefore, our model is closer to the ”alpha” model as described in [6]. In the alpha-model, the generated torque is directly controlled by the muscle activations producing the equilibrium point trajectories and adapting the stiffness. Following our simple model of muscle, in our model, the generated torques depend on both the activation of muscles (i.e. their stiffness) and on the muscle lengths. Our model has also a major difference with the alpha-model as it associates muscle activations with learned visuomotor configurations instead of relying on a temporal sequence of muscle activations. In the next section, we explain how the muscle activations are learned and associated with the recruited visuomotor categories in order to allow motor control.

4.2 Categorization of proprioceptive and visual space

The DM-PerAc model can use the previously described simplified muscle model with learned visuomotor associations to build a visuomotor controller (Fig. 3). Visual and proprioceptive signals are merged into visuomotor categories which are associated to the muscular activations determining the arm movements i.e. defining postural attractors. First, we present how the visual and proprioceptive categories are learned and computed. In the next section, we will present how the visuomotor categories are built from the learned visual and proprioceptive categories. We will also detail how the postural attractors are learned as muscle activations associated with the visuomotor categories. Both processes occur alternatively and participate to the sensorimotor babbling process allowing the robot to learn how to act.

Proprioceptive categories are recruited during a sensorimotor exploration process. Considering the agonist/antagonist muscles, the proprioceptive information is defined by the angular positions of the controlled joints $\mathbf{P} = [\theta_1^+ \dots \theta_N^+ \ \theta_1^- \dots \theta_N^-]$ (index m[3]). Each value $\theta^{+/-}$ is positive and normalized with respect to the agonist or antagonist references (see Fig. 2). The categorization of the proprioceptive input is described by (9) and (10). The proprioceptive inputs **P** are encoded into categories $\mathbf{S}^P$ with Gaussian responses depending on a variance parameter β^P. The variance parameter β^P enables to increase or to reduce the selectivity of the sensory categories. They are recruited with a process based on Adaptive Resonance Theory [14]. If the current input **P** is too different from any encoded sensory pattern $\mathbf{W}_i^P$, i.e. if the recognition $\mathbf{S}_i^P$ is under a vigilance threshold λ^P, then a new category i_r is recruited ($\varepsilon^P = 1$). The current sensory input **P** is stored on the weights $\mathbf{W}_{i_r}^P$ to the i_r^{th} category. Even though a slow adaptation of the encoded categories is also possible, we do not consider it in this article.

$$\begin{cases} S_i^P = exp(-\frac{\sum_m (P_m - W_{im}^P)^2}{2\beta^P}) \\ \Delta W_{i_r j}^P = \varepsilon^P \cdot (P_m - W_{i_r m}^P) \\ \text{with } \varepsilon^P = \mathscr{H}(\lambda^P - \max_i(S_i^P)) \end{cases} \tag{9}$$

with the Heaviside function $\mathscr{H}(x) = 1$ if $x > 0$ and 0 otherwise. The recognition activities $\mathbf{S}^P$ are normalized to give the output of the recognition process $\mathbf{R}^P$ (10).

$$R_i^P = \frac{S_i^P}{\sum S^P} \tag{10}$$

The output R_i^P can be interpreted as the probability that the sensory category i is the current sensory state of the robot. In practice, we approximated the sensory categorization process to a winner-takes-all which corresponds to the variance parameter β^P tending to 0 i.e. the selectivity for the categories R_i^P is maximal.

In our robotic setup, the visual information is captured by a single camera. A visual feature detector (e.g. color detector) enables to extract points of interest. The information is then projected over two 1D fields or vectors using population coding. Each vector codes the accumulated salience for the projected points of interest. The retina-centered vectors

[3] Bold letters indicates vectors whereas plain letters are scalars.

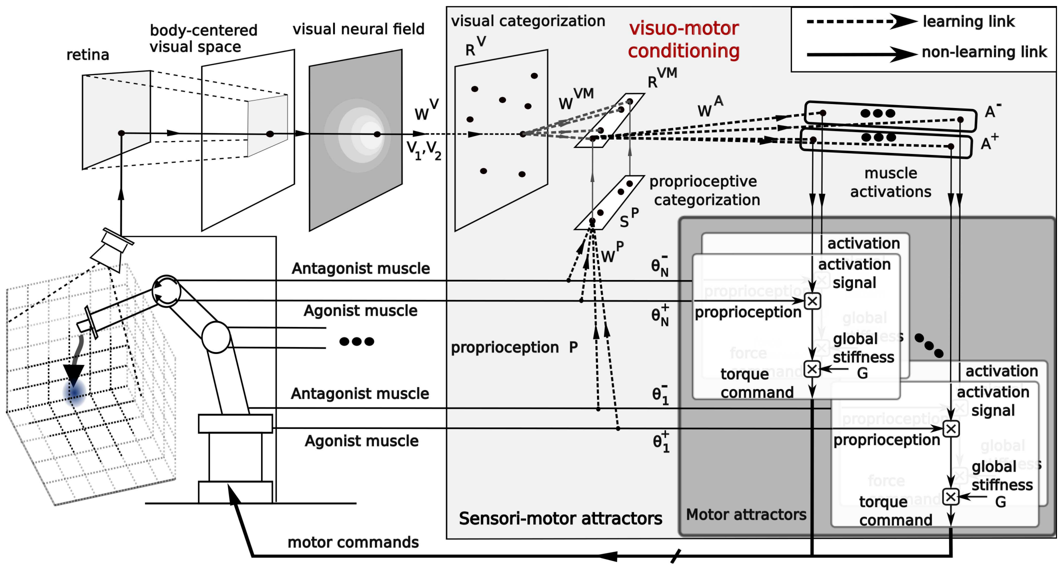

Fig. 3 Architecture of the visuomotor arm controller. Both visual and proprioceptive information are categorized. The visual input is associated with the proprioceptive input. The visuomotor categories are then associated with the muscle activations defining the motor attractors. The visual input activates the associated visuomotor categories and thus the corresponding motor attractors.

are then converted into body-centered vectors by a transformation using the pan and tilt angles of the camera. The body-centered vectors are computed as Dynamic Neural Fields [60]. Thus, they exhibit bifurcation and memory properties which are interesting in this attentional processing context. The coordinates (v_1, v_2) of the maximally salient point in this field are considered as the visual input. The visual categories are updated and learned using the equations (11) based on the equations (9).

$$\begin{cases} R_k^V = \frac{S_k^V}{\sum S^V} \text{ with } S_k^V = exp(-\frac{\sum_l (V_l - W_{kl}^V)^2}{2\beta^V}) \\ \Delta W_{k_r l}^V = \varepsilon^V \cdot (V_l - W_{k_r l}^V) \\ \text{ with } \varepsilon^V = \mathscr{H}(\lambda^V - \max_k(S_k^V)) \end{cases} \tag{11}$$

The recruitment of a visual category R_k^V increases the vigilance threshold λ^P of the proprioceptive categorization in order to facilitate the recruitment of a proprioceptive category if none already encodes the current posture.

4.3 Associating learned visuomotor categories with muscle activations

The visual and proprioceptive signals are merged in a visuomotor layer. There is a bijection between the proprioceptive categories and the visuomotor categories. Whenever a new proprioceptive category is recruited, a new visuomotor category S_i^{VM} is also recruited and associated with it. The visuomotor category is then associated with the muscle activations **A** maintaining the categorized posture. The aim of the visuomotor learning process is to determine which visual category R_k^V is maximally activated when the arm reaches the attractor posture S_i^P. The connection weights W_{ik}^{VM} are increased depending on the co-activated visual (R_k^V) and proprioceptive (S_i^P) categories (12):

$$\Delta W_{ik}^{VM} = \varepsilon^{VM} \cdot S_i^P \cdot (f(S_i^P) \cdot f(R_k^V) - W_{ik}^{VM}) \tag{12}$$

with ε^{VM} a constant learning rate. The function f is defined by $f(X_l) = 1$ if $X_l = \max_l(X_l)$ and $f(X_l) = 0$ otherwise. The co-activation is only learned when the arm is close enough to the posture S_i^P, so the learning is modulated by the factor S_i^P that checks if the similarity measure S_i^P is high enough. Incorrect visuomotor associations can be progressively forgotten.
The activities of the neurons in the visuomotor layer are computed with the following equations (13):

$$\begin{cases} R_i^{VM} = \frac{S_i^{VM}}{\sum S^{VM}} \quad \text{with} \quad S_i^{VM} = R_i^P \cdot \sum_k (g(W_{ik}^{VM}) \cdot R_k^V) \\ g(W_{ik}^{VM}) = \quad \begin{matrix} 1 \text{ if } \left(\frac{W_{ik}^{VM}}{\max_k(W_{ik}^{VM})} \right)^n > 0.5 \\ 0 \text{ otherwise} \end{matrix} \end{cases} \tag{13}$$

A weight W_{ik}^{VM} contributes either as a factor 1 or 0 in the update equation. The connection with maximal weight, among the input connections to a neuron i, always gives a factor equal to 1. Other connections can be “active” (factor equal to 1) if their weights are close enough to the maximum. Several visual categories can then activate the same visuomotor category. The normalization of the activities of the visual categories R_k^V ensures that the activities of the visuomotor categories S^{VM} are always smaller than 1. The saturation of the neural activities is thus avoided. Besides, when the exponent n tends to $+\infty$ only the connection with maximal weight

is equal to 1 and any others are null. We consider this particular case in the experiments.
The learning is performed on-line and fast. It is also incremental. By modifying some parameters (vigilance λ^P/λ^V or variance β^P/β^V) of the sensory categorization process, new visual and proprioceptive categories can be added on-line and are directly available for the visuomotor control. The vigilance parameter determines how much categories can overlap. Increasing the vigilance, i.e. allowing more overlapping, will increase the number of recruited categories. The variance parameter of the Gaussian kernels can be decreased with a similar result. If the variance is reduced, the selectivity of the categories increases and more categories will be recruited. Maintaining the vigilance level enables to maintain a certain level of overlapping and thus of interferences during learning.

As a result of a visuomotor association learning, a visual input can elicit visuomotor categories which activate motor actions (muscle activations) to drive the arm to the proprioceptive configuration associated with the visual constraint. When a new visuomotor category is recruited, the muscle activations enabling to maintain the visuomotor configuration (in practice maintaining the proprioceptive configuration is enough) are learned. Muscle activation coefficients are learned on-line in a perception-action process. The sensory-motor loop is essential. As the system acts, it corrects or modifies its motor commands on-line to maintain the desired posture of the arm. The corrective movements are learned by increasing the adequate connection weights to the muscle activation neurons $\mathbf{A} = [A_1 \dots A_{2N}] = [\mathbf{A}^+, \mathbf{A}^-]$. The activities of the visuomotor categories $\mathbf{R}^{VM}$ determine the muscle activations $\mathbf{A}$ with (14):

$$A_m = \sum_i W^A_{mi} \cdot R^{VM}_i \tag{14}$$

where the weight W^A_{mi} is the learned activation of m^{th} muscle to maintain the arm in the proprioceptive configuration i. In order to learn the muscle activations, the proprioceptive configuration corresponding to a recruited visuomotor category is stored. This proprioceptive signal $\hat{P}$ is then used as a supervision for the muscle activation learning. The desired position $\hat{P}_m$ is learned in one shot by associating $\mathbf{P}$ to $R^{VM}_{i_r}$ when the i^{th}_r visuomotor category is recruited. The corresponding update and learning equations are (15):

$$\hat{P}_m = \sum_i W^{\hat{P}}_{mi} \cdot R^{VM}_i \quad \text{with} \quad W^{\hat{P}}_{mi_r} = P_m - W^{\hat{P}}_{mi_r} \tag{15}$$

During the muscle activation learning process, the system selects a visuomotor configuration that is to be learned (for instance the last recruited visuomotor category i_r). The robot tries to reach the visuomotor configuration using the associated proprioceptive configuration $\hat{P}$ to correct movements. This selection induces that only the target visuomotor configuration is active (with i_r the selected configuration, $R^{VM}_{i_r} =$

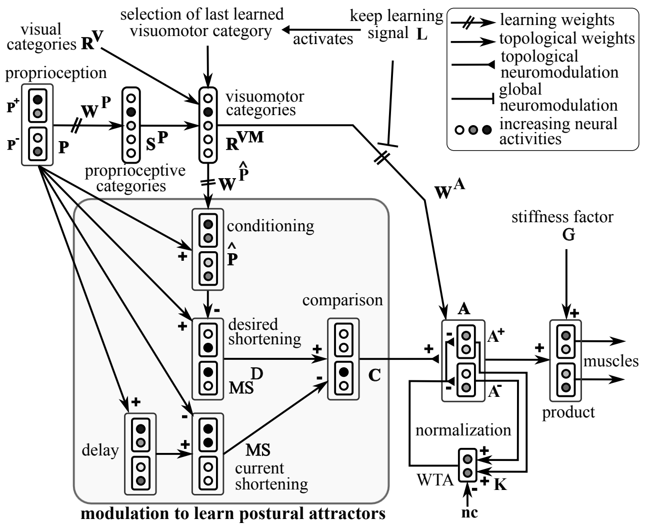

Fig. 4 Neural network learning the muscle activations to maintain the robotic arm in desired proprioceptive configurations. Learning is based on a neuromodulation process increasing the weights W^A_{mi} so the muscle activations **A** enable to maintain the desired posture. A second neuromodulation loop induces the normalization of the stiffness **K** of the different joints to avoid saturating the muscle activations.

1 and $R^{VM}_{i \neq i_r} = 0$), so only the corresponding weights W^A are modified. When the system learns the muscle activations, no other visuomotor category can be learned, the visuomotor exploration is suspended. The exploration resumes when the motor control meets the condition (no more correction). The learning equation (16) is based on a positive and a negative term and one learning factor:

$$\begin{aligned}\Delta W^A_{mi} = \mathscr{H}(L - th_L) \cdot (\ & \varepsilon^A \cdot C_m \cdot R^{VM}_i \cdot (1 - W^A_{mi}) \\ & -\alpha^A \cdot W^A_{mi} \cdot \max_j [K_j - nc]^+)\end{aligned} \tag{16}$$

where ε^A is a learning rate, α^A is a decay rate and $[x]^+ = x$ if $x > 0$ and 0 otherwise. The positive term in (16) increases the muscle activations thus changes the attractor so that the equilibrium posture matches the desired posture $\hat{P}$. This adaptation is based on the correction signal C detailed below (17). The role of the negative term in (16) is to normalize the stiffness K_j of the joints j to the constant value nc[4]. As the negative term changes all muscle activations with the same factor α^A, it does not modify the equilibrium posture, only the stiffness is modified. This normalization process is necessary to avoid the saturation of both the weights W^A_{mi} and the neural activities A_m which would prevent any further correction of the movements.
The part of the architecture in the gray rectangle in Figure 4 is dedicated to the computation of the correction signal C. For each joint, the signal C compares the desired movements $\mathbf{M}^D$ with the current movements $\mathbf{M}$ (17) to determine if a muscle should contract more i.e. if the muscle activations

[4] In practice, the range of activities was $[0,1]$ and we used $nc = 0.1$

associated to the target visuomotor configuration should be increased.

$$C_m = \mathscr{H}(M_m^D - M_m) \tag{17}$$

Each neuron in the desired movement layer M^D evaluates the need to contract the muscle m ($M_m^D = 1$ or 0) to correct the posture. To do so, the equation of M_m^D (18) determines if the muscle "length" P_m (i.e. θ^+ or θ^-) should be reduced to match the desired "length" $\hat{P}_m$.

$$M_m^D = \mathscr{H}(P_m - \hat{P}_m - th_D) \tag{18}$$

where th_D is a threshold under which no correction is requested. It defines the accuracy constraint for the movements. The correction signal C_m (17) does not change the muscle activations if the current movement M_m already reduces the muscle length i.e. if P_m is decreasing. This condition allows to avoid overshooting the correction of the movements. This condition is computed by $M_m(t) = \mathscr{H}(P_m(t-\Delta t) - P_m(t))$ with $M_m = 1$ when no change of the muscle activation should occur.
The learning factor ($\mathscr{H}(L - th_L)$) induces that muscle activations are learned during a variable period of time depending on the comparison between the "learning enabling" signal L and the threshold th_L. This signal L evaluates the need to continue adapting the muscle activations (19).

$$\begin{aligned} L(t) = [&\mathscr{H}(L(t-\Delta t) - th_L) \cdot \sum_m [C_m - \hat{C}_m]^+ \\ &+ \gamma^L \cdot L(t-\Delta t) + t_g(t)]^+ \end{aligned} \tag{19}$$

In our implementation, the learning is triggered ($t_g(t) = 1$; 0 otherwise) when a new visuomotor category i_r is recruited. Therefore, the muscle activations are directly learned after the recruitment of each visuomotor category ensuring that motor commands are associated to all visuomotor categories. Yet, the muscle activation learning may also be triggered by other signals, like a random signal arbitrarily selecting categories to refine the associated motor command. The muscle activation learning continues as long as there is unexpected correction of the muscle activations. Such unexpected correction is determined by comparing for each muscle occurring correction C_m with its prediction $\hat{C}_m$. The occurrence of an unexpected correction increases the value of the signal L, thus extending the learning time period. The forgetting factor γ^L modulates the time period during which no unexpected corrections must occur before the attractor adaptation ends. The prediction $\hat{\mathbf{C}}$ of the corrections is learned by conditioning with $\mathbf{C}$ the unconditional stimulus and $\mathbf{R}^{VM}$ the conditional stimulus (20).

$$\hat{C}_m = \sum_i W_{mi}^C \cdot R_i^{VM} \text{ with } \Delta W_{mi_r}^C = \varepsilon^C \cdot R_{i_r}^{VM} \cdot (C_m - \hat{C}_m) \tag{20}$$

The learning rate ε^C is small to have a memory effect. The learned muscle activations are expected to maintain the arm close to the postural target, so no more corrections are necessary. The learning of this posture can then stop and the motor exploration resumes. Sometimes, the arm may be blocked by an obstacle (possibly itself). The current version of the architecture does not include an obstacle avoidance process (still, a security module can block movements to prevent damages), so the muscles may only be more and more contracted without correcting the position. The deadlock is broken when the prediction $\hat{\mathbf{C}}$ of the continuous correction finally compensates the detected correction C and stops the learning. The motor exploration can then resume and the muscle activations related to this unsuccessfully learned postural attractor, is not used for the control. Interestingly, in [51], the authors hypothesized that the role of dopamine could also be to detect novelty and maintain or repeat recent actions providing the adequate context for learning. In our case, detecting unpredicted situations (corrections) can maintain the learning of a given posture instead of resuming the motor exploration.

As mentioned above, the weights W_{mi}^A and the muscle activations $\mathbf{A}$ are bounded ($\mathbf{A} \in [0,1]^N$) due to the learning rule (16). Hence, the muscle activations $\mathbf{A}$ are multiplied by a constant stiffness factor G increasing the amplitude of the apparent stiffness. The resulting equilibrium point is unchanged whereas the apparent stiffness is now equal to $G \cdot K$. The previous dynamic equation (5) becomes (21):

$$\begin{aligned} I_j \cdot \ddot{\theta}_j = G \cdot (A_j^+ \cdot (\theta_{j,max} - \theta_j) - A_j^- \cdot (\theta_j - \theta_{j,min})) \\ -\sigma_j \cdot \dot{\theta}_j + \tau_{j,e} + \eta_j \end{aligned} \tag{21}$$

For each joint j, a noise term η_j is also added in the motor command producing varying exploratory movements to help the learning of the muscle activations.

5 Experimental results

5.1 Postural attractor learning

The process to learn postural attractors was tested and validated in a simulation[5] of the Katana arm used in our robotics experiments (Fig. 5 and 6). In this experiment, the external torque τ_e was null. As the arm moves, the muscle activations are increased so that each joint is maintained at the desired position (Fig. 5). The progressive adaptation of the muscle activations depends on random movements (7). Still, the arm finally stabilizes at the desired posture (Fig. 6). As the muscle activations increase, the shifts of the equilibrium point due to learning become smaller and smaller. This property results from the ratio in the equation of the equilibrium point ($\theta_{j,eq} = \frac{A_j^+}{A_j^+ + A_j^-}$). So, the equilibrium position converges to the desired position while the stiffness

[5] with the software Webots (Cyberbotics)

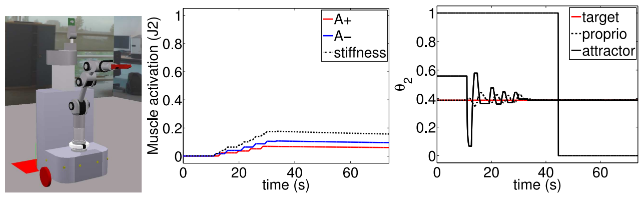

Fig. 5 Webots simulation of a Katana arm. Learning a postural attractor in the 4 DOF motor space. The evolution of the muscle activation and of the resulting equilibrium point is given for the 2nd articulation of the arm. A uniform random noise ($[-0.5, 0.5]$) is added to the torque command. When the movement of a joint is in the direction opposite to the target direction, the corresponding muscle activation is increased. As the stiffness increases, the shift of the position of the equilibrium point at each correction becomes smaller enabling to perform a gradient descent toward the target position. Besides, a bigger stiffness increases the robustness to the noise.

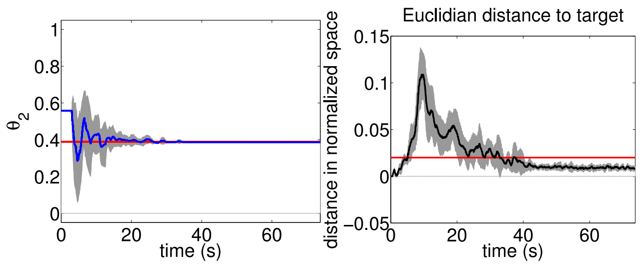

Fig. 6 The attractor learning test is reproduced 10 times. Left: Mean position of the learned attractor for joint 2 with the limits of the gray area representing the standard deviation. Right: Average and standard deviation of Euclidean target distance in the normalized joint space. The red line is the distance constraint th_D for each joint proprioception. The mean distance to the target decreases down to this constraint.

($K_j = A_j^+ + A_j^-$) increases. The behavior adaptation is quite slow because of the low frequency of the hardware control loop of the Katana arm (about 7 Hz). Another major constraint is the speed encoding in the robot arm firmware. Very low speed is not available because of the discretization of the values. Instead of an unnatural freezing of movements when the speed should be very close to null, the articulations keep rotating at the fixed minimal speed. These small oscillations give in fact a more natural aspect to the idle movements of the arm. The feeling of a frozen system is avoided during human robot interaction. In this experiment, there was no external torque. The reason is that the servo-controllers of the Katana electrical robotic arm is not compatible with external perturbations. This is a strong limitation of the hardware. We performed simulations to show that our model can also manage this case.

5.2 Maintaining a particular posture under external torque

In order to show that our model can also cope with external torques, we use a simple simulation of a 1D arm (Fig. 7a). First, the muscle activations are learned in the case of a gravitational torque (Fig. 7b-c). In the equation of control (21) the external torque is the following gravitational torque $\tau_e = -ma * g * le * sin(\theta)$ with the mass ma, the gravity constant $g = 9.81$ and the length le between the rotational axis and the gravity center. In order to compensate this torque, the muscle opposing to the gravity contributes more to maintain the posture (Fig. 7c). This solution is more energy efficient and accurate than simply increasing the overall impedance. It corresponds to the change of reciprocal activation level observable in human motor behaviors in equivalent circumstances [22]. The movements resulting from the learn controller are shown in Figure 7d. Figure 7e shows that the error made is indeed below the accuracy threshold used during learning.

We also tested the impact of increasing the noise level of η_j (in (21)) which corresponds to stochastic perturbations of the movements. If the controller was learned with a low noise level, the movements are strongly perturbed by the noise. The position error while maintaining the learned posture has a strong variance (Fig. 7f). Then, the postural attractor was learned with the increased noise level (Fig. 7g-h). As a result, the muscle activations are also increased which corresponds to increasing the stiffness (Fig. 7h). So, the produced movements are less perturbed by the noise (Fig. 7i-j). Our model can learn how to maintain a posture control under a gravitational torque, and it can also increase the stiffness of the movement to resist to stochastic perturbations during learning.

5.3 DM-PerAc visuomotor controller

We validated the visuomotor controller in the same 3D simulation of a Katana robot arm as in previous section. In Figure 8(a-c), the robot performs a motor babbling with parameters inducing a low selectivity and thus a very low level of accuracy for the recruited visual and proprioceptive states. A simple test to evaluate the visuomotor learning is to reproduce a trajectory given in the visual space. A star shaped trajectory is given as visual input to the system (Fig. 8b). The trajectory resulting from the visual processing of the arm end effector tracking is displayed. The robot tries to follow the trajectory but because of its sparse learning, the performance is very limited. In the developmental process of the robot, the parameters determining the sparsity of learning may be changed to recruit more visual and proprioceptive categories (Fig. 8(d-f)). The new visuomotor attractors are integrated on-line to the initial learning. The performance of the system is increased. Figure 9 displays the visual trajectories of the desired and real position of the arm end effector. The mean square error is shown with the mean error and the standard deviation to compare the evolution of the performance with the inclusion of more attractors. The same kind of performance could have been obtained by directly

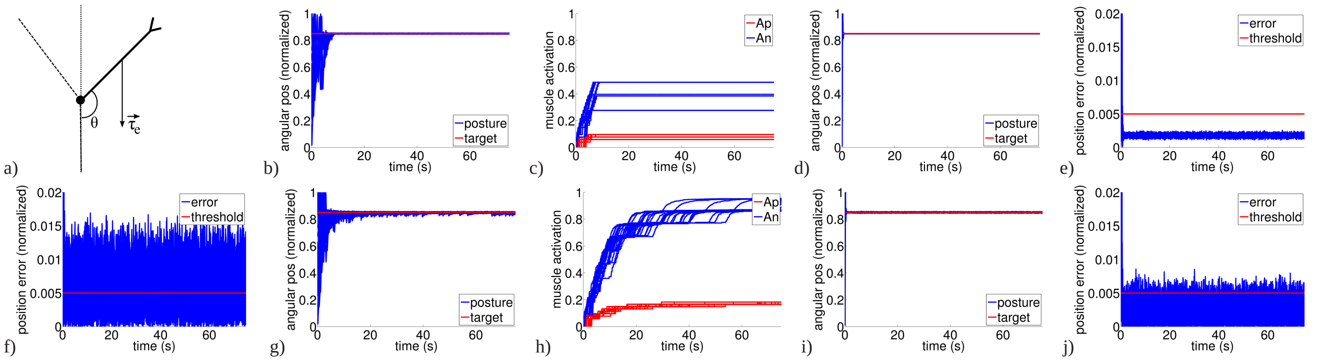

Fig. 7 a) A simple 1D model of an arm is used to test muscle activation learning under gravitational torque. The parameters are $g = 9.81$, $ma = 2kg$ and $le = 0.4m$. The angle θ is normalized with respect to the movement range $[0, 5\pi/4]$. b) Trajectories for 30 samples of posture control learning. c) Evolution of the muscle activations during learning. The muscle opposing to the gravity contributes much more than the other one. d) 10 examples of trajectories produced by one of the learned posture controllers. e) Corresponding position error with respect to the target (0.85). f) The noise level η_j is increased (from 0.1 to 1.5). The movements are then less accurate. g-h) The posture control is learned like in b-c) but with the increased noise level $\eta_j = 1.5$. i-j) As a result, the accuracy in reaching the target position is improved (lower variance).

learning with the parameters increasing the selectivity of the coding.

To sum up figures 8 and 9, learning a postural attractor takes time, and learning many attractors will slow the exploration of the whole motor space, but provide a better coding resolution and therefore a more accurate trajectory. Thus, very accurate trajectories could only be reproduced at the cost of a longer exploration and learning phase. In previous studies [3] we have simulated with the PerAc model that the learning time of all the possible sensorimotor associations of a 6DOF model of the Katana robotic Arm with a high resolution CCD camera would require hundreds of thousands of movements. Taking a mean approximation of the time necessary to perform one simple movement with our mechanical robot, we have calculated that the whole exploration and learning of all the possible categories would require more than three years (without optimization). Such amount of time is still applicable to the DM-PerAc model, since the number of possible categories (if we consider purely the maximal amount possible) is similar. Of course, such a computation is a caricature, since the creation of categories is by definition a mean to avoid systematic learning.
Nevertheless, several considerations lead us to think that such algorithms are consistent with the developmental course of a human baby :

– this time course (several years), taken as an order of magnitude is acceptable, compared to the time needed to develop the coordination of the whole human body (even if we limit to the coordination of one arm or one hand). We just have to refer to the time needed to master some movements in sports such as a golf swing, or the time to learn to write. Progressive learning is still present after months or years.
– if the maximal learning time is very long, DM-PerAC allow a very fast learning of simple trajectories with 10 to 20 attractors. The robot can hence perform simple tasks even if with limited accuracy. Such fast acquisition of coarse and elementary actions is crucial in term of behavior and is consistent with developmental psychology : coarse actions support early imitation to communicate before the age of 9 months [9], or object grasping before the age of 9 months [46], and of course early sensory-motor exploration before the first year [29].
– In addition of these elementary actions, the DM-PerAc model can let the category creation continue in order to improve the capabilities of the robot. New visual and proprioceptive categories can be recruited while the motor babbling is resumed. Therefore, the robot can continuously evaluate the co-occurring proprioceptive and visual inputs to improve its visuomotor model with the newly learned categories. The visuomotor associations can be progressively updated as the system continues its babbling.
– Altogether, these characteristics allow to speculate about when the babbling should stop. We can formulate the hypothesis that the visuo-motor babbling goes on while the agent has not received remarkable repeated feedback. The feedback could be purely “physical” (for example a tactile sensory motor contingency, for example when an object is grasped) or “social” (the expression of a caregiver) and modulate the strength of the learning. Thus, fast coarse actions and long progressive learning can be complementary in a global progress loop.

Interestingly, classical developmental psychology studies also observe that such progress loop are guided by the cephalocaudal (the more the limbs are far from the head, the later they are available and mature to be implied in actions) and the proximo-distal (the more the articulation are far from the root of the limb, the later they are available and mature to be implied in actions) laws. These laws reflect constraint of the body development that imposes a step by step process of the motor control. One of the consequence of this scheme is to constrain a coarse to fine learning where each change in the child's development result in an increasingly refined level of skill development [56].

In [16], several regression algorithms (including LWPR [63]) were compared on the visuomotor control learning and performance. The evaluation task is target tracking by the arm end effector of a robot. The system must produce the movements to reach a target given by its visual position, thus the learned inverse kinematic models are compared. A stereo camera detects the target, and its 3D Cartesian position is computed. In most of the tests, the target follows a star shaped trajectory path in a vertical plane. The regression algorithms learn a forward kinematic model in order to perform the tracking, thus focusing the exploration process on the motor space to perform the task. The forward model allows to estimate the Jacobian matrix of the kinematic model, the inversion of this matrix and the 3D position of the target provide the motor control of the robotic arm. In this article, we have tested the DM-PerAc visuomotor controller on tracking a target moving on a star shaped trajectory. In our experiment protocol, the visuomotor learning is open-ended. Also, the target coordinates are simulated (no occlusion) in the 2D visual space. The trajectories after learning are comparable to those obtained in [16]. Still, the regression techniques produce smoother trajectories more accurate at the points of the star path. However, inverting the Jacobian matrix requires a specific processing in order to avoid singularities. Such a matrix inversion is not satisfying in the perspective of the developmental approach and is also difficult to model as a biologically plausible process.

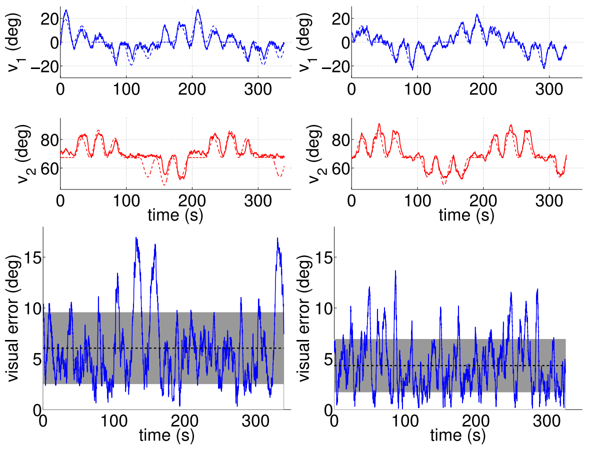

Fig. 9 Comparison between the trajectories from initial (Left hand column) and consecutive learning (Right hand column). Initial learning: mean error 6.0 degrees, standard deviation 3.5 degrees. Consecutive learning: mean error 4.3 degrees, standard deviation 2.5 degrees.

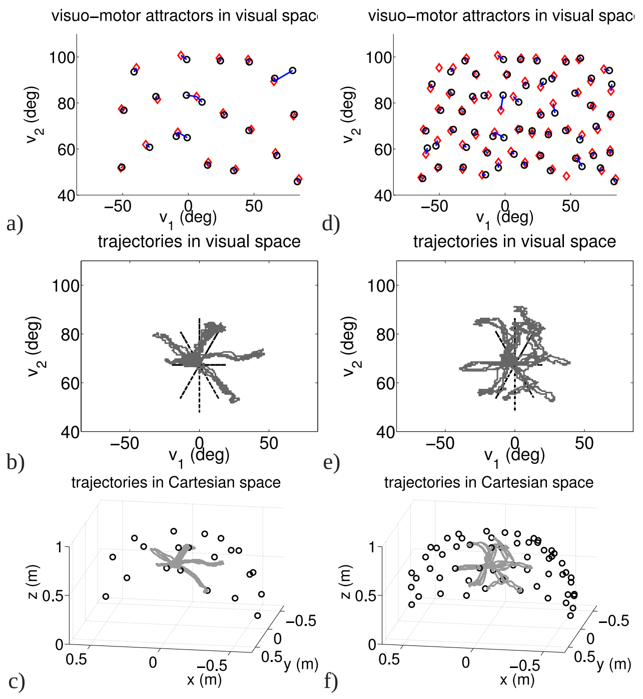

Fig. 8 Simulation of on line learning and adaptation of sensorimotor attractors with a 4 DOF arm and a 2D camera. Left hand column presents the results after an initial sparse learning and the right hand column gives the results after learning continued with learning parameters inducing more selectivity in the state recruitment. *a)* During the motor babbling, the robot recruits visual states (red diamonds) and proprioceptive states (black circles). Each proprioceptive state is associated with one visual state (blue link). *b)* After learning, the visual input is artificially switched to a star shaped trajectory in the visual space (dark line). According to the visual state recognition, the robot moves so the arm end effector trajectory tries to follow the visual input (gray dashed-line). *c)* Movements performed in the 3D Cartesian space during the star shaped trajectory reproduction. *d)* As the parameters changes, the robot can complete its previous learning by recruiting more visual states and proprioceptive states. *e)* The movements of the arm matches more closely to the star shaped trajectory in the visual space. *f)* Corresponding movements in the 3D Cartesian space.

5.4 Bifurcation property of the DM-PerAc controller

We compare the properties of the DM-PerAc controller with the properties of the Dynamic Neural Field based controller. Dynamic Neural Fields (DNF) based on Amari equation [2] are a solution to motor control used to navigate [60], [31] or to control a robotic arm [39, 3]. Biological studies showed that the activations of some neurons in the motor cortex are correlated with the direction of the movement to be performed [28]. In DNF, the activity profile of the field takes the shape of a Gaussian centered on the input stimuli. Besides, the derivative of the activity profile can provide the dynamics of the control [60]. Dynamic Neural Fields have interesting dynamical properties: memory to filter non stable or noisy stimuli and bifurcation capabilities enabling reliable and coherent decision when multiple stimuli are presented.

In Figure 10, we show that (i) the trajectories generated by the DM-PerAc model can be analyzed and integrated to

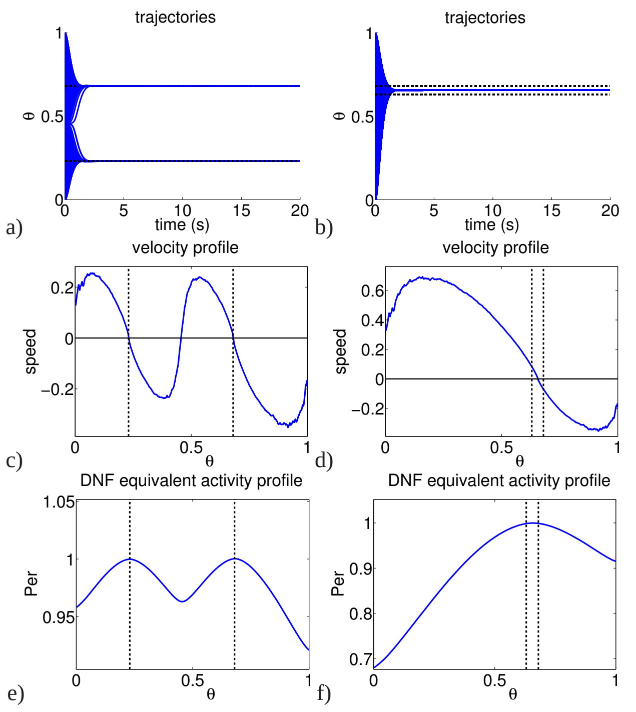

Fig. 10 Bifurcation capabilities in the DM-PerAc controller. Top row (a-b) shows the trajectories (blue lines) and the two learned attractors (black dashed lines). The middle row (c-d) displays the angular velocity profiles in function of the proprioception θ. The bottom row (e-d) gives the perception activity profile equivalent to the activities in a Dynamic Neural Field. In left hand column, the learned attractors are distinct whereas in the right hand one they are closer resulting in one merged behavioral attractor.

build the DNF equivalent profile of activity, and (ii) there are bifurcation capabilities in our controller. In our tests, the state space is $[0,1]$. Trajectories generated by the DM-PerAc controller are averaged into the actions $Ac(\theta)$ depending on the state of the system (position). In practice, $Ac(\theta)$ is discretized into a vector with components that are the values for different θ. The result is thus the velocity profile given in Fig 10c and d. In [48], we proposed that the action Ac is the derivative of a potential function defining the perception of the system. The action Ac is thus spatially integrated to obtain the perception Per (22).

$$\forall k, Per_k = \int_{[0,k/n]} Ac(\theta)d\theta + cste \tag{22}$$

where Per is a vector of dimension n with components equal to the integration of the action Ac at different positions $\theta = k/n$. The integration constant cst is chosen so the maximal component value of Per is equal to 1. The perception profile Per is equivalent to the activity profile of a DNF, and shows bifurcation properties (see Fig 10). The DM-PerAc model can produce behaviors similar to those obtained with the use of an explicit DNF without the need to define the whole field. However, the property of memory is not directly available in the model, but some others processes could complete the DM-PerAc architecture to obtain this property.

6 Use and extensions of the DM-PerAc model

6.1 Encoding trajectories with the DM-PerAc controller

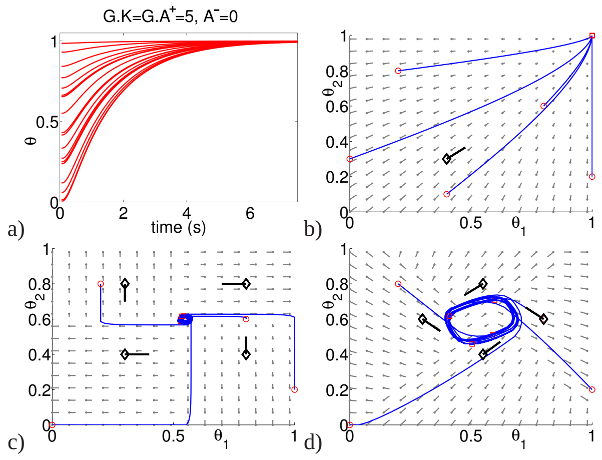

Fig. 11 *a)* Trajectories in 1D space with an asymetric muscle activation pattern (a muscle is inactive). Trajectories start from different random positions. Activation signals are $G \cdot A^+ = G \cdot K = 5$, $A^- = 0$. The control parameters are $\sigma = 5$, $\Delta t = 0.05$ and the moment of inertia $I = 1$. *b-c)* Attraction basins in a bounded 2D space $[0,1]^2$ with DM-PerAc model. Given the learned position/movement couples (black diamonds, thick black lines), a force field is generated (small gray points and lines). For each joint, only one of the agonist/antagonist muscles is activated like in a). Initial (circle) and final (square) points of the trajectories are indicated. *b)* Vector field corresponding to one learned proprioception/activation couple. *c)d)* Four state/action couples are learned. Four trajectories with different starting points are represented in the 2D state space. With only four couples, the system can learn a loop trajectory. The size of the loop depends on the speed thus is related to the damping σ and the stiffness K. *c)* $\sigma = 10, G \cdot K = 10$. *d)* $\sigma = 5, G \cdot K = 10$. The other parameters of the system are the time increment $\Delta t = 0.05$ and the moment of inertia $I = 1$.

It is possible to use the learned postural attractors in a time based sequence with the attractors that are successively and transiently activated. This process was used in the work described in Section 6.2. However, the DM-PerAc architecture is not limited to using this kind of trajectory coding. Now, we consider the case where only one of the muscles around a joint is activated (activation different of 0) while the other one is inactive. This configuration of activation signals induces movement toward the extreme limit of joint (full flexion or extension) (Fig. 11a). At the lower level of motor control, the muscle activations can be either interpreted as defining a postural attractor or as defining locally the movement to be performed (orientation and strength). As explained in Section 3, such associations between sensory categories and actions can define trajectories. The studied task is simply to reproduce a loop in the 2D motor space. Among four encoded states, each of them are associated with two 1D controllers i.e. four muscle activation coefficients each. The muscle activations correspond to the demonstrated direction of movement. For each joint, only one of the muscle activation (agonist or antagonist) is different of null. An example of a vector field in 2D space defined by one state/action couple is given in Fig. 11b. An attraction basin can effectively be generated (Fig. 11c and d). The trajectories in the 2D state space show that the stiffness K and the damping σ control the movement speed and thus can change the size of the loop. Trajectories could be encoded using the low level state/muscle activation associations. This coding can thus be a basis for both posture and trajectory encoding. In the next section, we will focus on learning stable postural attractors.

6.2 Imitative behaviors with the DM-PerAc controller

The visuomotor controller based on the DM-PerAc model can be used for the emergence of low level imitative behaviors and can even be a basis for deferred imitation. An arm controller, based on learning visuomotor associations, can let low level imitation emerge [3]. In a first phase of babbling, the robot learns its body schema as multiple associations between the visual position of its arm end effector and the joint configuration of its arm. If the robot visual perception is enough limited (using only movement information or the detection of colored patches), the robot can look at the hand of an interacting human and still believes it is its own hand. According to the previously learned visuomotor associations, this situation can induce an incoherence between the visual information from the teacher's hand and the motor information from the hand of the robot. As the controller is a homeostat, it tends to maintain equilibrium between the visual and the motor signals. Thus, the robot tries to reduce the visuomotor incoherence by moving its hand to match the visual input. Low level imitation emerges as the movements of the robot follow the movements of the human (Fig. 12). In the next stage of development of the robot, this low level visuomotor controller can be the basis for learning from observation. We consider that the learning robot can now memorize the sequence of the visual positions demonstrated by the teacher while it is inhibiting its own movement [52]. Then, as the robot internally rehearses the encoded visual sequence, the predicted visual position of the next state can be given to the low level visuomotor controller. The robot reproduces the demonstrated sequence of gestures according to what was perceived during the demonstration. The robot is capable of doing some deferred imitation [52, 53].

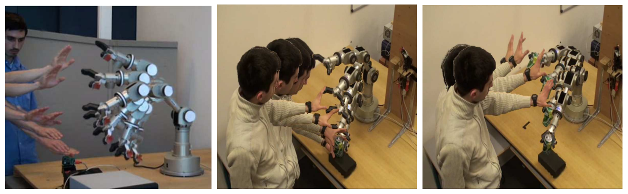
Fig. 12 Example of imitation behaviors. *Left* : Low level imitation of meaningless gestures. Qualitative comparison of imitated gestures performed in front of the robot. Perception ambiguity and a homeostatic controller induce movements to maintain perceptual equilibrium. The robot performs low level imitation of directly observed gestures. *Middle and right* : Gesture imitation can be used to bring the arm end-effector toward objects (here, to grasp a can) or interesting parts of the environment. It can become a common basis for learning by observation and learning by doing.

6.3 Attractor selection and visuomotor control refining

The refining potential of the DM-PerAc model can be enhanced by the "Yuragi" (fluctuations) based attractor selection model [23] which relies on the following Langevin's equation (23):

$$\Lambda \cdot \dot{\mathbf{x}} = \xi \cdot f(\mathbf{x}) + \eta \tag{23}$$

where Λ is a time constant, the vector $\mathbf{x}$ describes the state of the system and the function f is the dynamics of the attractor selection model. The main constraint that this attractor function f must respect is to define attractors. For instance, the function f can simply derive from a potential function with attractor points. Other particular examples of definitions of the function f can be found in [23, 52]. When the coefficient ξ is big, the term $\xi \cdot f(\mathbf{x})$ predominates. The state of the system converges to one of the attractors defined by f. Feedback on the current movement performance modulates the coefficient ξ. The feedback gives more influence to the attractor function f or to the stochastic exploration term η. As a result, the system can switch from exploration between the different known attractors to exploitation of the closest attractors. According to the feedback, the function f can be adapted so that some attractors are shifted toward the desired positions. Thus, the desired positions can be learned.
The principle of muscle activation learning (Sec. 4.3) in DM-PerAc is quite similar. The first difference is that the function f depends on the muscle contraction. During muscle activation learning, only one visuomotor category is active so only one postural attractor is active. The exploration is partly due to the noise on the motor command and also to the oscillations of the arm (when the stiffness is still low). During learning, the muscle activations are changed so that the resulting attractor is effectively shifted toward the desired position. So, this process can be seen as a low level use of the "Yuragi" principle.

The "Yuragi" principle can also be use in DM-PerAc when all the visuomotor categories are available. The movement dynamics is influenced by all the attractors associated

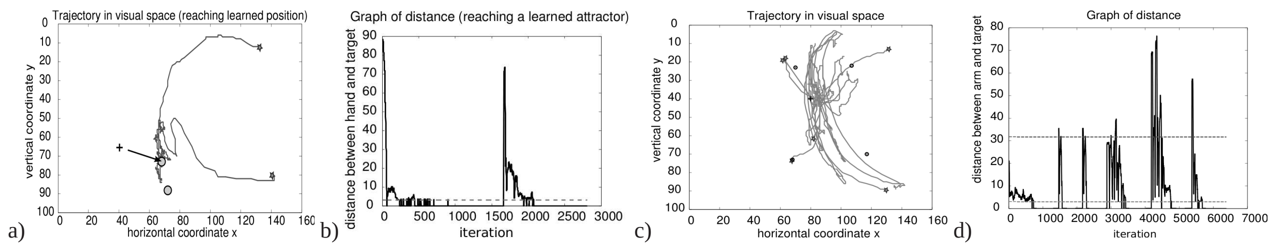

Fig. 13 Visual target reaching with a visuomotor controller using the "Yuragi" principle. The feedback is based on the target distance in the visual space. A known attractor can match the target (a,b) or the target can be between learned attractors (c,d). *(a,c)* Trajectories of the robot arm end effector in the visual space. The black circles correspond to the learned attractors and the black cross is the visual target to be reached. The stars are the starting positions for each trial. *(b,d)* Evolution of the distance between the arm end effector and the target in the visual space (number of pixels). Dark gray dash-line shows the average distance to the attractors. The light gray line shows the threshold under which the target is reached. *a*: Trajectories while reaching a learned attractor, 2 attractors activated, 2 trials with different starting positions. *b*: Corresponding evolution of the target distance. *c*: Trajectories while reaching a not previously learned position, 4 attractors activated, 6 trials with different starting positions. *d*: Corresponding evolution of the target distance. In both cases, the arm end effector reaches the target, although, when it is not a learned position, the reaching can be quite long due to the random exploration.

to these categories and activated by visual and proprioceptive information. In that case, the "Yuragi" principle allows to improve the accuracy of the movements. In Figure 13, we tested the reaching of a visual position using the "Yuragi" principle [52]. The robot arm end effector reaches the visual target both when it is near the visual position of a learned attractor (Fig. 13(a-b)) and when it is between the learned attractors (Fig. 13(c-d)). While performing tasks, the robot can use the "Yuragi" principle to reach targets which were not explicitly learned as attractors. When necessary, a new attractor could be recruited to learn how to reach a target that would otherwise be long to reach. The performance of the visuomotor controller could be improved for particular cases without recruiting many useless attractors.

7 Conclusion-discussion

Our previous works enabled to explain trajectory learning (PerAc model [26]) and imitative behaviors [3]. Even though these different works have in common the sensorimotor learning principle, their properties could not directly be combined due to motor control issues. We propose the Dynamic Muscle PerAc (DM-PerAc) model to control a robot arm with multiple DOF (Sec. 4). It combines the principles of the PerAc model with a simple model of agonist/antagonist muscles where the muscle activations determine the movements of the robotic arm. The low level motor control is equivalent to impedance control. The DM-PerAc model can incrementally learn on-line the visuomotor control of the robot arm. During a motor babbling process, proprioceptive and visual categories are recruited and associated together (kinematic model) depending on co-activation. The DM-PerAc model then learns the postural attractors associated with the visuomotor categories to define the visuomotor control. Trajectories can also be coded by combining state/action couples like in the PerAc model (Sec. 6.1). The states are associated with asymmetric muscle activations to generate movements in particular directions. In section 6.2, we showed that imitative behaviors can be obtained with the DM-PerAc visuomotor controller. This controller can also be a basis for higher level encoding and imitation behaviors.

Until now, we mainly experimented the DM-PerAc model on a Katana robotic arm. However, the hardware of this robotic device is limited for impedance control. In particular, the servo-controller of the Katana arm does not allow to manage external perturbations like gravitational torque. In Section 5.2, we showed in a simple 1D arm simulation that the DM-PerAc model can accurately learn a postural attractor under a gravitational torque. However, the impedance control was learned instead of performing an on-line adaptation to perturbations. In future work, the adaptation process will be added to the model. Also, in future work, we will exploit the full potential of the DM-PerAc model to control movements of a hydraulic torso robot called TINO[6]. This robot was developed with the aim of allowing physical interaction and compliance. Impedance control is fully compatible with this hardware. With the DM-PerAc model, the visuomotor controller of the robot TINO can be learned. Besides, the DM-PerAc model is also a good basis to study imitative behaviors and interaction.

In this article, the motor control is based on a spring based model of muscles ; however, we do not pretend that modifying the stiffness of these spring-like muscles corresponds to an accurate model of neuro-muscular control. The rest-length of the muscles, motor reflexes and other physiological properties are also important. Still, the aim of the DM-PerAc model is to allow sensorimotor dynamics learning with the generated behaviors that can be either attractor postures or trajectory following. Using muscle activations

[6] The robot TINO was co-funded by the french projects INTERACT and SESAME TINO, the Robotex and the CNRS. The robot only recently arrived in ETIS lab.

has the advantage to make learning easier whatever the dynamics is (postural attractor or trajectory).

The computational cost of the DM-PerAc visuomotor controller can be reduced in different ways. The neurons corresponding to categories (visual, proprioceptive, visuomotor) not yet recruited can be ignored in the neural update process. Also, the number of visual to visuomotor links (W^{VM}) may be reduced by using some lists of links dynamically managed according to the recognition of the visual and proprioceptive categories. This solution would allow to use far fewer links than if considering the whole set of visual to visuomotor links.

We gave solutions to learn attractor points as they are used in the visuomotor controller for imitation behaviors. The learning of trajectories or paths is not described in this article. In the DM-PerAc model, postural attractors can be used as via-points to encode trajectories and we used this kind of solution in deferred imitation [52]. However, a correct encoding of dynamic trajectories should rely on state/action couples defining attraction basins, like in the PerAc model (Sec. 3). The advantage is that agonist and antagonist muscles would not need to be active at the same time. The stiffness and the energy consumption can be reduced. In future work, we will study the activation patterns generated by this trajectory encoding model. In particular, we want to explore whether and how the state-action coding may allow the triphasic pattern of movement observed in humans [55].

Although we proved that the DM-PerAc model enables dynamical trajectory encoding, the learning of the adequate state/action couples is still an ongoing issue. In the PerAc model, the states and actions were associated by direct conditioning. The orientation to follow (action) could be estimated by integrating the followed orientation while moving. The orientation to follow could also be demonstrated to a passive robot. In the DM-PerAc model (Fig 14a), a direct conditioning is possible but a particular process is necessary to extract the unconditional stimulus from a passive demonstration. Changes of proprioception cannot be directly converted into muscle activations (for instance, the muscle activations must change to perform the same movement manipulating objects with different masses). The “Yuragi” idea (Sec. 6.3), adapted to the DM-PerAc model, can be a potential solution to this issue. We believe that “Yuragi” idea could allow to locally learn combinations of attractors defining not only postural attractors but also particular speed vectors. Still, the remaining issues are what the adequate feedback is and how it can be learned from a demonstration.
Finally, using the same encoding and the same kind of learning, the robot should be able to learn trajectories like in Fig.14b mixing posture attractors and trajectory shaping.

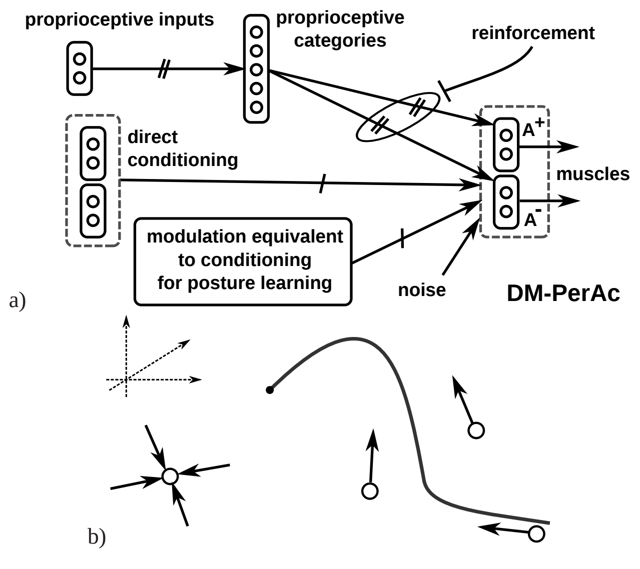
proprioceptive inputs
proprioceptive categories
reinforcement
direct conditioning
A+
A-
muscles
modulation equivalent to conditioning for posture learning
noise
DM-PerAc
a)
b)

Fig. 14 *a)* Possible solutions to learn muscle activations in the Dynamic Muscle PerAc model. b) Example of dynamic trajectory with postural attractors and trajectory shaping constraints. Both components can be coded similarly in the DM-PerAc architecture.

Appendices

A Summary of the parameters and equations used in the Dynamic-Muscle PerAc model

The different parameters and equations presented in this article are respectively summarized in Table 1 and Table 2.

The proprioceptive (visual) categorization depends on the vigilance parameter λ^P (λ^V) and the parameter β^P (β^V) of the Gaussian similarity measure. High vigilance values would imply that recruited categories overlap. We use $\lambda^P = \lambda^V = 0.005$ to avoid interferences between categories. The values of the Gaussian parameters are very low so the categories are selective enough. During the learning step, different values are used to increase progressively the number of learned categories ($\beta^P = 0.002$ then $\beta^P = 0.001$, and $\beta^V = 2 \cdot 10^{-4}$ then $\beta^V = 5 \cdot 10^{-5}$). During the tests, the vision must drives the movements, so the proprioceptive categories must be less selective than the visual categories ($\beta^P = 0.1$ and $\beta^V = 5 \cdot 10^{-5}$).
In the experiments, the muscle activation learning depends on the learning factor $\varepsilon^A = 10^{-3}$ and the decay factor $\alpha^A = 10^{-4}$. As the learning factor is small, the stiffness K_j of each joint changes slowly. Still, the equilibrium position is rapidly adapted because it depends on the ratio of the muscle activations. Also, the decay must be slow enough to allow the learning. With an error threshold $th_D = 0.01$, the muscle activations around a joint are adapted if the position error is over a hundredth of the rotational range.
The parameters th_L and γ^L define the dynamics of the “learning enable” signal L i.e. determine the amount of time to learn each postural attractor. The used values are $th_L = 10^{-5}$ and $\gamma^L = 0.95$ so the motor exploration resumes after a time period of about 10 seconds without correction of the movements.

Acknowledgements This work was supported by the INTERACT French project referenced ANR-09-CORD-014.

Table 1 DM-PerAc Model: parameter summary with values used in experiments for the open parameters

$\mathbf{A}$	$=$	$[A_1, \ldots, A_{2N}]$ muscle activation (stiffness)
$\mathbf{A}^+$, $\mathbf{A}^-$	:	activation of agonist $(^+)$ and antagonist $(^+)$ muscles for each joint ($\mathbf{A} = [\mathbf{A}^+, \mathbf{A}^-]$)
$\mathbf{C}$	:	comparison of desired and current movements, determines the need to correct muscle activations, modulates the increase of $\mathbf{W}^A_{mi}$
$\hat{\mathbf{C}}$	:	prediction of $\mathbf{C}$ for a given visuomotor category i in $\mathbf{R}^{VM}$
G	:	stiffness factor, counterbalancing the bounded muscle activations $\mathbf{A}$ (ex: $G = 60$)
$\mathbf{K}$	:	stiffness
i, i_m, i_r	:	indexes of proprioceptive category, winning proprioceptive category and next recruited proprioceptive category
$\mathbf{I}$	:	moment of inertia (ex: $I = 1$)
j	:	index of joint
k, k_m, k_r	:	indexes of visual category, winning visual category and next recruited visual category
l	:	visual coordinates
L	:	attractor learning signal
m	:	index of muscle
$\mathbf{M}^D$, $\mathbf{M}$	:	desired muscle shortening, current muscle shortening
n	:	exponent, used in the update of the visuomotor categories (ex: $n = 100$)
N	:	number of joints
$\mathbf{R}^P$, $\mathbf{R}^V$,$\mathbf{R}^{VM}$	:	normalized activities of $\mathbf{S}^P$,$\mathbf{S}^V$ and $\mathbf{S}^{VM}$
$\mathbf{P}$	$=$	$[P_1 \ldots P_{2N}] = [\mathbf{P}^+ \mathbf{P}^-]$ proprioceptive input
$\mathbf{P}^+$, $\mathbf{P}^-$	:	agonist and antagonist proprioceptive input $[\theta_1^+ \theta_2^+ \ldots]$, $[\theta_1^- \theta_2^- \ldots]$
$\mathbf{S}^P$, $\mathbf{S}^V$	:	recognition activities of proprioceptive and visual categories respectively
$\mathbf{S}^{VM}$	:	visuomotor category, merging visual and proprioceptive signals
$t, t - \Delta t$	:	current time step, previous time step
th_D	:	threshold on target distance to estimate desired movement (ex: $th_D = 0.01$)
th_L	:	threshold on L under which current attractor learning is stopped (ex: $th_L = 10^{-5}$)
$\mathbf{V}$	:	visual input (coordinates in visual field)
$\mathbf{W}^P_{im}$, $\mathbf{W}^V_{kl}$	:	learning weights to proprioceptive ($\mathbf{S}^P$) or visual ($\mathbf{S}^V$) categories
$\mathbf{W}^C_{mi}$	:	learning weights to $\hat{\mathbf{C}}$
$\mathbf{W}^A_{mi}$	:	learning weights to $\mathbf{A}$
$\mathbf{W}^{VM}_{ik}$	:	learning weights to $\mathbf{R}^{VM}$
α^A	:	decay factor of muscle activation learning ($\mathbf{W}^A_{mi}$) (ex: $\alpha^A = 10^{-4}$)
β^P, β^V	:	variance parameter of the Gaussian kernels of proprioceptive P or visual V categories.
ε^A	:	learning factor of muscle activation ($\mathbf{A}$) learning (ex: $\varepsilon^A = 10^{-3}$)
ε^C	:	learning factor of the predictor of $\mathbf{C}$ (ex: $\varepsilon^C = 0.2$)
ε^P, ε^V	:	learning factor of proprioceptive P or visual V categorizations.
γ^L	:	forgetting factor of the attractor learning signal L (ex: $\gamma^L = 0.95$)
λ^P, λ^V	:	vigilance of proprioceptive categorization P or visual categorization V. (ex: $\lambda^P = \lambda^V = 0.05$)
σ_j	:	damping (ex: $\sigma_j = 11$)
θ_j, $\dot{\theta}_j$, $\ddot{\theta}_j$	:	rotation angle of a joint, velocity, acceleration
θ_j^+, θ_j^-	:	positive angular value measured in the agonist or antagonist reference (see Fig. 2)
$\theta_{j,max}$, $\theta_{j,min}$	:	maximal and minimal angular value of a joint
$\theta_{j,eq}$	:	equilibrium point resulting from muscle activations
τ_j, τ_e	:	rotational torque, external torque

General tools

Heaviside function:	$\mathscr{H}(x) = 1$ if $x > 0$, 0 otherwise
Kronecker symbol:	$\delta_{ij} = 1$ if $i = j$, 0 otherwise
	$[x]^+ = x$ if $x > 0$, 0 otherwise

References

1. Albu-Schäffer A, Ott C, Hirzinger G (2007) A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. Int J Robot Res 26(1):23–39
2. Amari SI (1977) Dynamics of pattern formation in lateral-inhibition type neural fields. Biol Cybern 27(2):77–87
3. Andry P, Gaussier P, Nadel J, Hirsbrunner B (2004) Learning invariant sensorimotor behaviors: a developmental approach to imitation mechanisms. Adapt Behav 12(2):117–140
4. Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Auton Syst 57(5):469–483
5. Atkeson CG, Andrew W, Stefan Schaal Z (1997) Locally weighted learning. In: Artif. Intell. Rev., pp 11–73
6. Bizzi E, Hogan N, Ivaldi FAM, Giszter S (1992) Does the nervous system use equilibrium-point control to guide single and multiple joint movements? Behavioral and Brain Sciences 15(Special Issue 04):603–613
7. Bullock D, Grossberg S (1989) VITE and FLETE: neural modules for trajectory formation and postural control. In: Hershberger W (ed) Volitional action, Adv. Psychol., vol 62, Elsevier, chap 11, pp 253–297

Table 2 DM-PerAc Model: equation summary

Motor control based on commands of stiffness of agonist/antagonist muscles around the joints j^a

$$\tau_j = A_j^+ \cdot \theta_j^+ - \sigma_j^+ \cdot \dot{\theta}_j^+ - (A_j^- \cdot \theta_j^- - \sigma_j^- \cdot \dot{\theta}_j^-)$$

Which is simplified from additional constraints (6) as:

$$\ddot{\theta}_j = \frac{K_j}{I_j} \cdot (\theta_{j,eq} - \theta_j) - \frac{\sigma_j}{I_j} \cdot \dot{\theta}_j \text{ with } K_j = A_j^+ + A_j^- \text{ and } \theta_{j,eq} = \frac{A_j^+}{A_j^+ + A_j^-}$$

Update and learning of the proprioceptive and visual categories
Proprioceptive categories (index i) based on the muscle proprioception $\mathbf{P} = [\theta_1^+, \theta_2^+ \dots, \theta_1^-, \theta_2^-, \dots]$ (index m):

$$R_i^P = \frac{S_i^P}{\sum S^P} \text{ with } S_i^P = exp(-\frac{\sum_m (P_m - W_{im}^P)^2}{2\beta^P})$$
$$\Delta W_{i_r m}^P = \varepsilon^P \cdot (P_m - W_{i_r m}^P) \text{ with } \varepsilon^P = \mathscr{H}(\lambda^P - \max_i(S_i^P))$$

Visual categories (index k):

$$R_k^V = \frac{S_k^V}{\sum S^V} \text{ with } S_k^V = exp(-\frac{\sum_l (V_l - W_{kl}^V)^2}{2\beta^V})$$
$$\Delta W_{k_r l}^V = \varepsilon^V \cdot (V_l - W_{k_r l}^V) \text{ with } \varepsilon^V = \mathscr{H}(\lambda^V - \max_k(S_k^V))$$

Visuomotor association learning

$$\Delta W_{ik}^{VM} = \varepsilon^{VM} \cdot S_i^P \cdot (f(S_i^P) \cdot f(R_k^V) - W_{ik}^{VM})$$
with $f(X_l) = 1$ if $X_l = \max_l(X_l)$ and 0 otherwise

Visuomotor categories update

$$\begin{cases} R_i^{VM} = \frac{S_i^{VM}}{\sum S^{VM}} \quad \text{with} \quad S_i^{VM} = R_i^P \cdot \sum_k (g(W_{ik}^{VM}) \cdot R_k^V) \\ \text{and } g(W_{ik}^{VM}) = 1 \text{ if } \left(\frac{W_{ik}^{VM}}{\max_k(W_{ik}^{VM})} \right)^n > 0.5 \text{ and 0 otherwise} \end{cases}$$

Postural attractor learning
Supervision signal based on incorrect movements:

$$\begin{cases} C_m = \mathscr{H}(M_m^D - M_m) \text{ where} \\ M_m^D = \mathscr{H}(P_m - \hat{P}_m - th_D) \text{ and } M_m(t) = \mathscr{H}(P_m(t - \Delta t) - P_m(t)) \\ \hat{P}_m = \sum_i W_{mi}^{\hat{P}} \cdot R_i^{VM} \text{ with} \\ W_{mi_r}^{\hat{P}} = \varepsilon^P \cdot (P_m - W_{mi_r}) \text{ (on recruitment of a new } R_{i_r}^{VM}) \end{cases}$$

Signal L indicating whether attractor learning should continue:

$$\begin{cases} L(t) = \mathscr{H}(L(t - \Delta t) - th_L) \cdot \sum_m [C_m - \hat{C}_m]^+ + \gamma^L \cdot L(t - \Delta t) \\ \hat{C}_m = \sum_i W_{mi}^C \cdot R_i^{VM} \text{ with } \Delta W_{mi_r}^C = \varepsilon^C \cdot R_{i_r}^{VM} \cdot (C_m - \hat{C}_m) \end{cases}$$

Muscle activation update and learning:

$$A_m = \sum_i W_{mi}^A \cdot R_i^{VM} \text{ with } \mathbf{A} = [A_1^+, A_2^+, \dots, A_1^-, A_2^-, \dots]$$

$$\Delta W_{mi}^A = \mathscr{H}(L - th_L) \cdot (\ \varepsilon^A \cdot C_m \cdot R_i^{VM} \cdot (1 - W_{mi}^A) - \alpha^A \cdot W_{mi}^A \cdot \max_j [K_j - nc]^+)$$

During attractor learning ($L > th_L$), after the recruitment of the visuomotor category i_r, a bias on the activation ensures that $R_{i_r}^{VM} = 1$ and $R_{i \neq i_r}^{VM} = 0$.

[a] The time step t is only written when a signal at different time step is used

8. Burdet E, Tee KP, Mareels I, Milner TE, Chew CM, Franklin DW, Osu R, Kawato M (2006) Stability and motor adaptation in human arm movements. Biol Cybern 94(1):20–32
9. Butterworth G (1999) Neonatal imitation: existence, mechanisms and motives. In: Nadel J, Butterworth G (eds) Imitation in Infancy, Cambridge: Cambridge University Press, pp 63–88
10. Calinon S, Guenter F, Billard A (2007) On learning, representing and generalizing a task in a humanoid robot. IEEE Trans Syst, Man, Cybern B Special issue on robot learning by observation, demonstration and imitation 37(2):286–298
11. Calinon S, D'halluin F, Caldwell DG, Billard A (2009) Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. In: Proc. of 2009 IEEE Int. Conf. on Humanoid Robots, pp 582–588
12. Calinon S, D'halluin F, Sauser E, Caldwell D, Billard A (2010) Learning and reproduction of gestures by imitation: an approach based on Hidden Markov Model and Gaussian Mixture Regression. IEEE Robot Autom Mag 17(2):44–54
13. Calinon S, Sardellitti I, Caldwell DG (2010) Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Taipei, Taiwan, pp 249–254
14. Carpenter GA, Grossberg S (2002) Adaptive resonance theory (ART). In: The handbook of brain theory and neural networks, MIT Press, Cambridge, MA, USA, pp 79–82
15. Chiaverini S, Siciliano B, Villani L (1999) A survey of robot interaction control schemes with experimental comparison. IEEE/ASME Trans Mechatronics 4(3):273–285
16. Droniou A, Ivaldi S, Padois V, Sigaud O (2012) Autonomous online learning of velocity kinematics on the iCub: a comparative study. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vilamoura, Portugal, pp 3577–3582
17. Feldman AG (1966) Functional tuning of the nervous system with control of movement or maintenance of a steady posture. II. Controllable parameters of the muscle. Biophysics 11(3):565–578
18. Feldman AG (1986) Once more on the equilibrium-point hypothesis (lambda model) for motor control. J Motor Behav 18(1):17–54
19. Feldman AG, Levin MF (2009) The Equilibrium-Point Hypothesis Past, present and future progress in motor control. In: Sternad D (ed) Progress in Motor Control, Advances in Experimental Medicine and Biology, vol 629, Springer US, book part (with own title) 38, pp 699–726
20. Flash T (1987) The control of hand equilibrium trajectories in multi-joint arm movements. Biol Cybern 57(4):257–274
21. Flash T, Hogan N (1985) The coordination of arm movements: an experimentally confirmed mathematical model. J Neurosci 5(7):1688–1703
22. Franklin DW, Burdet E, Tee KP, Osu R, Chew CM, Milner TE, Kawato M (2008) CNS learns stable, accurate, and efficient movements using a simple algorithm. J Neurosci 28(44):11,165–11,173
23. Fukuyori I, Nakamura Y, Matsumoto Y, Ishiguro H (2008) Flexible control mechanism for multi-DOF robotic arm based on biological fluctuation. From Anim Animat 10 pp 22–31
24. G C, L S (1968) The human eye-movement mechanism: experiments, modeling, and model testing. Arch Ophthalmol 79(4):428–436
25. Ganesh G, Albu-Schaffer A, Haruno M, Kawato M, Burdet E (2010) Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks. In: Robotics and Automation ICRA 2010 IEEE International Conference on (2010), IEEE, 2010 IEEE International Conference on Robotics and Automation, ICRA 2010, pp 2705–2711
26. Gaussier P, Zrehen S (1995) PerAc: a neural architecture to control artificial animals. Robot Auton Syst 16(2-4):291–320
27. Gaussier P, Moga S, Banquet JP, Quoy M (1998) From Perception-Action loops to imitation processes: a bottom-up approach of

learning by imitation. Appl Artif Intell 1(7):701–727
28. Georgopoulos A, Schwartz A, Kettner R (1986) Neuronal population coding of movement direction. Science 233(4771):1416–1419
29. Gergely G (2001) Is early differentiation of human action a precursor to the one-year-old's understanding of intentionality? Developmental Psychology 37:57982
30. Giovannangeli C, Gaussier P (2010) Interactive teaching for vision-based mobile robots: a sensory-motor approach. IEEE Trans Syst, Man, Cybern A 40(1):13–28
31. Giovannangeli C, Gaussier P, Désilles G (2006) Robust mapless outdoor vision-based navigation. In: IEEE/RSJ Int. Conf. on Intelligent Robots and systems, IEEE, Beijing, China
32. Hersch M, Billard A (2006) A biologically-inspired model of reaching movements. In: Proc. of the 2006 IEEE/RAS-EMBS Int. Conf. on Biomedical Robotics and Biomechatronics
33. Hill AV (1938) The heat of shortening and the dynamic constants of muscle. Proc R Soc B: Biol Sci 126(843):136–195
34. Hoffmann H, Pastor P, Park DH, Schaal S (2009) biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance. In: Proc. IEEE Int. Conf. Robot. (ICRA2009)
35. Hogan N (1984) An organizing principle for a class of voluntary movements. J Neurosci 4(11):2745–2754
36. Huxley AF (1957) Muscle structure and theories of contraction. Prog Biophys Biop Ch 7:255–318
37. Ijspeert AJ, Nakanishi J, Schaal S (2003) Learning attractor landscapes for learning motor primitives. In: Adv. in neural information processing systems 15, cambridge, ma: mit press, pp 1547–1554
38. Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P, Schaal S (2013) Dynamical movement primitives: learning attractor models for motor behaviors. Neural Comput 25(2):328–73
39. Iossifidis I, Schoner G (2004) Autonomous reaching and obstacle avoidance with the anthropomorphic arm of a robotic assistant using the attractor dynamics approach. In: Proc. IEEE Int. Conf. Robot. (ICRA2004), Inst. fur Neuroinformatik, Ruhr-Univ., Bochum, Germany, IEEE, vol 5, pp 4295–4300
40. Iossifidis I, Schoner G (2006) Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for an redundant robot arm. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'06), Institut fur Neuroinformatik, Ruhr-Universitat Bochum, pp 580–585
41. Jiménez-Fabián R, Verlinden O (2011) Review of control algorithms for robotic ankle systems in lower-limb orthoses, prostheses, and exoskeletons. Med Eng Phys 34(4):397–408
42. Klute GK, Czerniecki JM, Hannaford B (2002) Artificial muscles: actuators for biorobotic systems. Int J Robot Res 21(4):295–309
43. Kohonen T (1982) Analysis of a simple self-organizing process. Biol Cybern 44(2):135–140
44. Kronander K, Billard A (2012) Online learning of varying stiffness through physical Human-Robot interaction. In: Int Conf on Robotics and Automation, pp 1842–1849
45. Lagarde M, Andry P, Gaussier P, Boucenna S, Hafemeister L (2010) Proprioception and imitation: on the road to agent individuation. In: Sigaud O, Peters J (eds) From Motor Learning to Interaction Learning in Robots, vol 264, Springer Berlin Heidelberg, Berlin, Heidelberg, book part 3, pp 43–63
46. Law J, Shaw P, Earland K, Sheldon M, Lee MH (2014) A psychology based approach for longitudinal development in cognitive robotics. Frontiers in Neurorobotics 8
47. Lungarella M, Metta G, Pfeifer R, Sandini G (2003) Developmental robotics: a survey. Connect Sci 15(4):151–190
48. Maillard M, Gapenne O, Hafemeister L, Gaussier P (2005) Perception as a dynamical sensori-motor attraction basin. In: Capcarrre M, Freitas A, Bentley P, Johnson C, Timmis J (eds) Adv. in Artificial Life, Lecture Notes in Computer Science, vol 3630, Springer Berlin Heidelberg, pp 37–46
49. Miyamoto H, Kawato M (1998) A tennis serve and upswing learning robot based on bi-directional theory. Neural Networks 11(7-8):1331–1344
50. Nehaniv CL, Dautenhahn K (2002) The correspondence problem. In: Dautenhahn K, Nehaniv CL (eds) Imitation in animals and artifacts, MIT Press, Cambridge, MA, USA, pp 41–61
51. Redgrave P, Gurney K (2006) The short-latency dopamine signal: a role in discovering novel actions? Nat Rev Neurosci 7(12):967–975
52. de Rengervé A, Boucenna S, Andry P, Gaussier P (2010) Emergent imitative behavior on a robotic arm based on visuo-motor associative memories. In: IEEE/RSJ Int. Conf. on Intelligent Robots and systems (IROS'10), Taipei, Taiwan, pp 1754–1759
53. de Rengervé A, Andry P, Gaussier P (2013) A brain-based architecture toward action imitation and task learning by observation and demonstration. Front in Neurorobotics p in revision
54. Rozo L, Calinon S, Caldwell D, Jimenez P, Torras C, Jiménez P (2013) Learning collaborative impedance-based robot behaviors. In: AAAI Conf on Artificial Intelligence
55. Sanes JN, Jennings VA (1984) Centrally programmed patterns of muscle activity in voluntary motor behavior of humans. Exp Brain Res 54(1):23–32
56. Santrock JW (2005) A topical approach to life-span development (2nd ed.). Boston: McGraw-Hill
57. Schaal S (1997) Learning from demonstration. In: Adv. Neur. In., mit press, vol 9, pp 1040–1046
58. Schaal S (2006) Dynamic movement primitives - a framework for motor control in humans and humanoid robotics. In: Kimura H, Tsuchiya K, Ishiguro A, Witte H (eds) Adaptive Motion of Animals and Machines, Springer Tokyo, pp 261–280
59. Schaal S, Atkeson CG (1998) Constructive incremental learning from only local information. Neural Comput 10(8):2047–2084
60. Schöner G, Dose M, Engels C (1995) Dynamics of behavior: theory and applications for autonomous robot architectures. Robot Auton Syst 16(2-4):213–245
61. Slotine JJE (1988) Adaptive manipulator control: a case study. IEEE Trans Autom Control 33(11):995–1003
62. Todorov E (2007) Optimal control theory. In: Doya K (ed) Bayesian Brain: Probabilistic Approaches to Neural Coding, Applied Mathematical Sciences, MIT Press, chap 12, pp 269–298
63. Vijayakumar S, D'souza A, Schaal S (2005) Incremental online learning in high dimensions. Neural Comput 17(12):2602–2634
64. Winters JM, Stark L (1985) Analysis of fundamental human movement patterns through the use of in-depth antagonistic muscle models. IEEE Trans Bio-Med Eng 32(10):826–839
65. Winters JM, Stark L (1987) Muscle models: what is gained and what is lost by varying model complexity. Biol Cybern 55(6):403–420