2014 5th IEEE RAS & EMBS International Conference on
Biomedical Robotics and Biomechatronics (BioRob)
August 12-15, 2014. São Paulo, Brazil

Generalization of the Tacit Learning Controller Based on Periodic Tuning Functions

Vincent Berenz*⋆, Mitsuhiro Hayashibe†, Fady Alnajjar* and Shingo Shimoda*
*RIKEN Brain Science Institute / Toyota Collaboration Center, Nagoya, Japan
†INRIA, LIRMM Universite de Montpellier Sud de France

Abstract— **Living organisms are characterized by their smooth adaptability to environmental changes and their robustness against morphological modifications. To investigate the computational mechanisms behind such learning scheme, we proposed *tacit learning* as a novel learning method. In tacit learning, there are no clear distinctions between learning and motor control: learning is a simple accumulation process embedded in the controller. In previous work, tacit learning was applied with success to bipedal locomotion of a 36 DoF humanoid robot. In this paper, we generalize the structure of the controller such as applying adaptive integration to a wider range of systems and behaviors. This is achieved by applying the principle of tacit learning in a hierarchical fashion, in which the value of a virtual periodic dynamic variable is tuned for continuous adaptation. This resulting PD-PI (proportional-derivative periodic-integration) controller preserves the advantages of tacit learning that the controllers do not include any prior knowledge of the system in which they are embedded. It also shares with biological systems the property that control and adaptation progress without explicit distinction between them.**

I. INTRODUCTION

Living organisms are characterized by their smooth adaptability to enviromental changes and their robustness against morphological modifications, e.g. due to aging or changes in muscle mass. Tacit learning was introduced to model the learning aspect of such control strategy [1]. In tacit learning, there are no clear distinctions between learning and motion controls: learning is embedded in the controllers, which update themselves in a continuous manner during action. Furthermore, the learning strategy does not rely on the explicit minimization of a global score function. This distinguishes tacit learning from the learning and adaptation methods such as artificial neural-networks [2][3], reinforcement learning [4][5][6] or GA [7]. Controllers are independent, interacting one with another only through their relative effect on the environment. A higher-level control architecture that orchestrates the action of these controllers for achieving a given task is not used. The learning scheme of tacit learning is based on simple signal accumulation. Each controller tunes its own activity based on integration of errors. Such learning strategy also plays a significant role in biological systems, e.g. in the long term depression of cerebellum [8][9] or in immune systems [10]. Tacit learning has two clear advantages. First, no model of the system is used therefore high adaptability can be achieved. Second, it shares with the biological control system the property that behavior control and adaptation to the environment progress without explicit distinction between them. In previous papers, we applied tacit learning by using the output of the controller as the error to be integrated by the controller. The corresponding proportional-derivative-input integration (PD-II) controller is presented in Figure 1. Applying this controller to a 36 DoF bipedal robot pre-implemented only with a very crude walking reflex resulted in the emergence of a gait that was highly adapted to the environment [1][11][12]. Tacit learning by means of task-space feedback error information is also presented in [13].

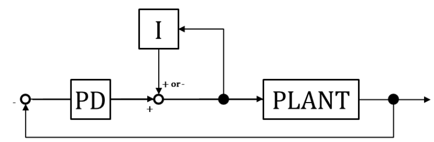


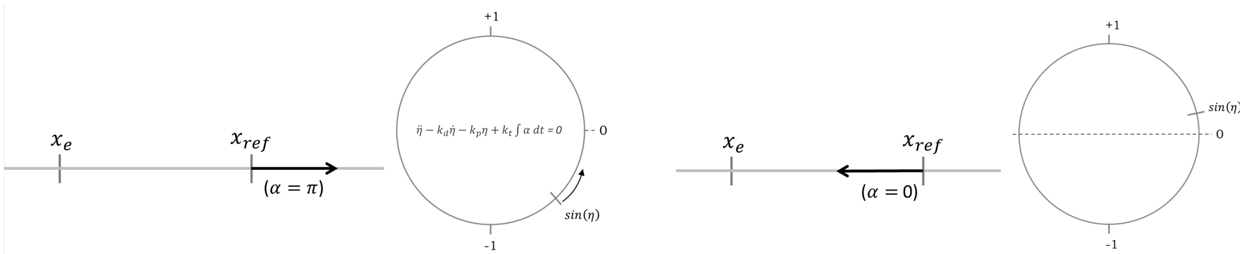
Fig. 1. Block diagram of the controller for tacit learning

In this paper, we present the use of tacit controllers based on the integration of errors expressed in artificial error spaces. This will allow to extend the applicability of tacit-learning while preserving its advantages, i.e. continuous adaptation of controllers. In the following section, we present tacit learning and the PD-II controller in details. In Section III, we investigate the applicability of tacit learning when integrating errors expressed in arbitrary error spaces. We show that periodicity in integration is required for continuous adaptation, resulting in a proportional-derivative-periodic-integration (PD-PI) controller that applies tacit learning in a hierarchical fashion. This controller is tested in Section IV: its adaptability is demonstrated as it allowed simulated arms of unknown characteristics to perform diverses tasks. Finally, the range of applicability of the controller is discussed. More specifically we consider use of abstract error spaces, such as a space defined by markers used for characterizing the status of motor-impaired patients.

II. PD-II CONTROLLERS

The general expression for a tacit controller is:

⋆ correspondance: vincent@brain.riken.jp



a. as the motion not in the required direction, the virtual system is updated

b. the virtual system stabilizes at a value suitable for task execution

Fig. 2. Representations of the Proportional-Derivative-Periodic-Integration Controller in a single dimensional error space. x_e represent a position in the control space corresponding to a zero error in the error space. On the right of the control space, a schematic representation of the virtual adaptive system is represented. Motion of the system in the undesired direction (i.e. $\alpha = \pi$) results in the update of the virtual system through integration of α. The virtual system stabilizes when the direction of the motion is corrected (i.e. $\alpha = 0$). A periodic function p is applied to the virtual variable η to unsure such stabilization occurs.

$$\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x_c} + f(\boldsymbol{q}) \tag{1}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{a} \tag{2}$$

$\boldsymbol{u}$ is the control, $\boldsymbol{x_c}$ the state variable expressed in the control space, $\boldsymbol{K}$ is the proportional and derivative gain matrices and $\boldsymbol{a}$ the effect to be minimized. f is a function $\mathbb{R}^n \rightarrow \mathbb{R}^n$ where n is the dimension of the vector $\boldsymbol{a}$. In this section we briefly present results previously obtained using $\boldsymbol{a} = \boldsymbol{u}$ and $f(\boldsymbol{q}) = \boldsymbol{k_t}^T\boldsymbol{q}$ where $\boldsymbol{k_t}$ is a vector of constant values. Detailed explanations as well as experimental results can be found in [12]. Proof of stability is presented in [14].

A 36 degrees of freedom (DOF) humanoid robot was considered as a $m + n$ rotative joints system. For the control of m joints, conventional PID controllers were used, applying trajectories of reference angles provided by the experimenter. These controllers implemented a crude walking motion unsuitable for the robot to walk with balance and rhythm. The remaining n joints were controlled by the PD-II controllers described in Figure 1 and equations (1) and (2). Signal accumulation by the integrator of these controllers due to the effect of gravity provided a process for behavior adaptation. After around 10 minutes of walking, the robot acquired balance. The experimental results demonstrated that that the walking gait that emerged was well adapted to the environment in terms of walking efficiency, rhythm adapatation and robustness toward the walking terrain.

These results can be briefly explained as follow. For a single joint, the PD-II controller can be written:

$$\tau = -k_p\theta + k_d\dot{\theta} + k_t \int \tau \mathrm{d}t \tag{3}$$

τ is the torque applied to the joint, θ the angle of the joint, k_p, k_d and k_t gain values. Motion of θ due to gravitational forces will results in a non-zero torque, which in turn will results in integration. Integration will stop and the controller stabilize for τ being zero, i.e. for the system being in a configuration in which gravitational effects are minimized.

This approach achieved adaptability:

- All joints are controlled independently and no model of the robot is used. Therefore the same controller could be used on robotic systems characterized by other kinematic parameters, i.e. the PD-II controller adapts to the system it controls.
- Environmental changes or change in hardware configuration will result in changes in gravitational effects and resume of the adaptive integration. Action and adaptation are continuous and not divided between "learning phases" and "action phases".

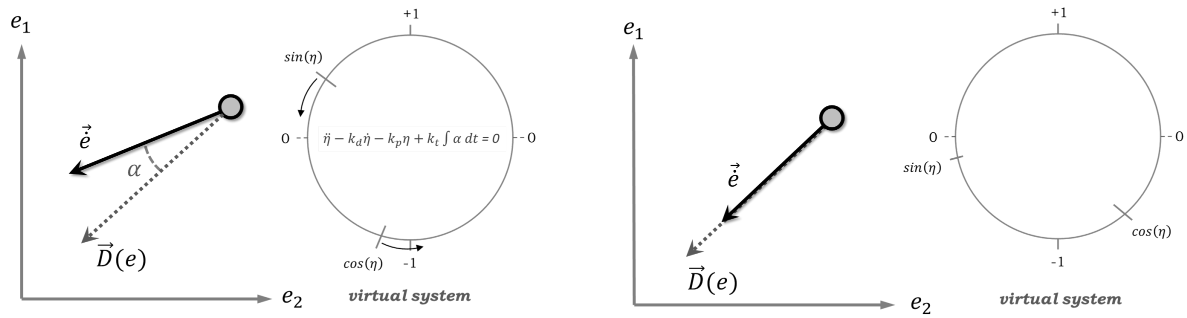
This approach could be used for other applications, but applicability of the PD-II controller is limited to the minimization of gravitational effects. In the following section, we investigate the use of tacit learning for minimization of other effects expressed in arbitrary error spaces.

III. PD-PI CONTROLLERS

In the previous section, the input to the system (the torque in the presented example) was used as the "error" to be minimized by the controller, i.e. integration would occur until the joint reaches a zero-torque configuration. In this section we present a tacit-learning based controller suitable for minimizing an error expressed in arbitrary error-spaces. The applicability of such controller is wide, as it could provide adaptability to sytems experiencing environmental or kinematic changes. A direct application, used for proof of concept and presented in Section IV, is the control of robotic arms of different kinematic configurations. But our final goal is the use of this controller using abstract error spaces, such as spaces defined by markers characterizing the status of post-stroke patients with the objective to develop novel robotic rehabilitation strategies. Thus, this section is deliberately unspecific in regard of the nature of the system and of the error space.

A. Task definition

We consider an arbitrary error vector space ξ in which the system is characterized by a position vector $\boldsymbol{e}$. We use the term "error space" rather than "task space" only to indicate



a. as the motion not in the required direction, the virtual system is updated

b. the virtual system stabilizes for suitable motion direction

Fig. 3. The Proportional-Derivative-Periodic-Integration Controller in a 2D error space. The position of the system in the error space is represented by grey points. $\dot{\boldsymbol{e}}$ is the velocity of the system in the error space and $\boldsymbol{D(e)}$ the direction of motion required for task completion. The virtual system is schematically represented on the right of the error space, and stabilizes for $\dot{\boldsymbol{e}}$ being colinear with $\boldsymbol{D(e)}$, i.e. the system performing the task.

that e being the zero vector is the desired position of the system once the task has been achieved.

We define a task as follow: A task is a function $D(\boldsymbol{e}) = \boldsymbol{d}$ where $\boldsymbol{d}$ is a unit vector expressed in the error space. The direction of $\boldsymbol{d}$ indicates the direction of motion suitable for task accomplishment. According to this definition, at a given time the robot is executing the task if α, the angle between $\boldsymbol{d}$ and $\dot{\boldsymbol{e}}$, is small. By specifying a succession of tasks, this approach is also suitable to define complete trajectory. A controller has one or two objectives:

- If the system is immobile in the error-space, the controller must induce the motion of the sytem.
- The controller must tune the motion of the system such as the system executes the task, i.e. the absolute value of α must be minimized.

To achieve adaptability, the controller must accomplish these objectives under the constraints that no model of the system is known; and that learning and control progress in parallel.

B. Controller design

As presented by equation (1) and (2), the form of the tacit controller for achieving the task under these contraints can be written:

$$\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x_c} + f(\boldsymbol{q}) \tag{4}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{e} \tag{5}$$

f can be described as the function that tunes the integration of the error components such as achieving an input u that induces a suitable motion in the system, i.e. $|\alpha| = 0$. In this paper we propose:

$$f(\boldsymbol{x}) = \boldsymbol{p}(\eta)\boldsymbol{a}^T(\boldsymbol{x}) \tag{6}$$

p is a periodic function and η a virtual scalar parameter. $\boldsymbol{p}(\eta)$ is defined as:

$$\boldsymbol{p}(\eta) = \left[\begin{array}{cccc} p_1(\eta) & p_2(\eta) & \cdots & p_n(\eta) \end{array}\right]^T \tag{7}$$

where each function $p_i(\eta)$ is periodic and of range $[-1, 1]$.

a is a function designed to ensure:

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \iff \boldsymbol{x} = \boldsymbol{0} \tag{8}$$

In the rest of the section, we explain this proposed design of f.

C. One dimension error space

The control strategy described by equations (4) to (8) is directly applicable for the control of two degrees of freedom (DoF) systems and two dimensional error spaces. Application to higher dimensional spaces is presented in Section III-E. But for clarity in explanation, we first present its application on a single DoF system and a one dimensional error space.

In a one dimension error space, $a(e) = k_t$, where k_t is a constant positive value, is suitable for ensuring condition (8). Sinus is also a suitabled function for p. In a one dimensional error space, α can be of only two values, 0 and π. The system represented by equations (4), (5) and (6) can be written:

$$u = -k_p x - k_d \dot{x} + k_t \int sin(\eta) e \mathrm{d}t \tag{9}$$

which is equivalent to:

$$u = k_p(x_{ref} - x) - k_d \dot{x} \tag{10}$$

$$x_{ref} = \frac{k_t}{k_p} \int sin(\eta) e \mathrm{d}t \tag{11}$$

Equation (10) is equivalent to a PD controller with x_{ref} as reference. Under this perspective, the role of integration is to find dynamically a reference suitable for obtaining a motion characterized by $|\alpha| = 0$. Assuming suitable gain values for k_p and k_d; and considering a value x_e in the task space for which the error in zero, using x_e as reference in the PD controller would result in achievement of the task. In robotics, inverse kinematic is usually used to compute such value of x_e. But because of the assumption that no model of the system is known, possible values of x_e can not be calculated using such analytical method. But it can be noted that integration in equation (11) results in a continuous

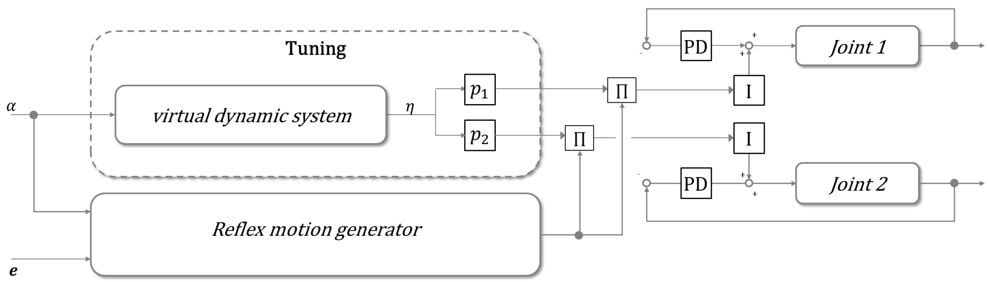


Fig. 4. Outline of the proposed approach. The reflex motion generator provokes a reflex motion which direction is tuned toward task execution via a tacit learning controller implemented in a virtual dynamic system.

change in the value of x_{ref}. If x_e and $sin(\eta)$ are of the same sign, this integration will result in a changes of x_{ref} such that motion of the system is performed in a suitable direction, i.e. α is zero. Similarly, if x_e and $sin(\eta)$ are of opposite signs, motion will be performed in the opposite direction, i.e. α is π. Therefore, η must be tuned such as x_e and $sin(\eta)$ are of the same sign. As the sign of x_e is unknown, we propose to model η as a virtual dynamic variable which values updates itself when α is not zero:

$$k_p\eta - k_d\dot{\eta} + \ddot{\eta} + k_t \int \alpha \mathrm{d}t = 0 \tag{12}$$

As a result, if α is not zero, integration in equation (12) results in a change in the value of η. The periodicity of sinus (equation 7) ensures that this integration will ultimately result in a change in the sign of $sin(\eta)$, therefore in a motion of the system characterized by $\alpha = 0$ (Figure 2). If α is zero, integration in equation (12) stops and η stabilizes at a suitable value for task execution. Periodicity of p is the crucial feature that insures that the virtual system will stabilizes at a suitable value of η.

Periodic integration occuring in the proposed PD-PI controller (Proportional-Derivative Periodic-Integration) results in the continuously tuning of the controller such as correcting the motion of the system into one corresponding to task accomplishment. Integration in the virtual dynamic system (equation 12) is the adaptive system that updates the controller. Adaptation is achieved in the sense that the controller continuously updates itself via integration when the system performs a motion that does not correspond to task execution.

D. Two dimensional error space

The system described in the previous section is directly applicable to 2DoF systems and 2D error spaces. The corresponding schematic representation is shown in Figure 3. Contrary to one dimensional spaces in which α was in $\{0, \pi\}$, in 2D space α is in $[-\pi, \pi]$. In a single dimensional space, a function $a(e) = k_t$, where k_t is a non-zero value, was suitable. For higher dimensions, the virtual system of equation (12) might result in a value of η such as $\boldsymbol{p(\eta)e}$ is the zero vector. Thus, to enforce condition (8), the use of more complex function is required for $\boldsymbol{a(e)}$. For example:

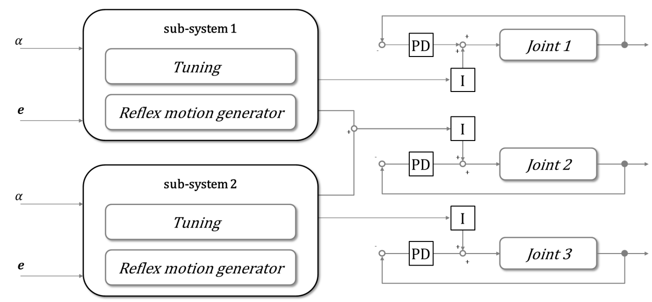


Fig. 5. Two subsystems are associated to accomplish a task of higher dimensionality

$$\boldsymbol{a(e)} = \left[\begin{array}{cccc} r(\boldsymbol{e}) & r(\boldsymbol{e}) & \cdots & r(\boldsymbol{e}) \end{array} \right] \tag{13}$$

$$r(\boldsymbol{e}) = rot(\phi + \pi/2)\boldsymbol{e} \cdot \boldsymbol{e} \tag{14}$$

'$\cdot$' is the dot-product, $rot(\phi + \pi/2)$ is a rotation matrice and ϕ is a non-zero angle. For p, the following periodic function can be used:

$$\boldsymbol{p(\eta)} = \left[\begin{array}{cc} sin(\eta) & cos(\eta) \end{array} \right] \tag{15}$$

Figure 4 presents the block diagram corresponding to the proposed controller applied to a 2DoF system. Application of the function a corresponds to the reflex motion generator as it insures motion of the system if the error is not the zero vector (equation 8). The resulting motion is characterized by a velocity $\dot{\boldsymbol{e}}$ which has an error angle α with the desired velocity. The controller is tuned via the virtual dynamic system such as minimizing $|\alpha|$, resulting in task accomplishment.

E. Extension to higher dimension

The controller presented in Figure 4 can be applied only to a 2DoF robot and on a 2 dimensional error-space. It can be extended to robots of higher DoF and error-space of higher dimensionality by treating the robotic system as a collection of 2DOF sub-systems, each associated with a tuning module and reflex motion generator module. Equations (4) and (5) are applied for each sub-system using the projection of the error position in a 2D subspace of the error-space. A single DoF can be controlled by several sub-systems. An example of such a system is provided in Figure 5, in which three DoF are controlled by two sub-systems which co-adapt. Such a controller is used in the following experimental section.

IV. EXPERIMENTAL RESULTS

A. Purpose and setup

In this section we validate that the proposed controller can adapt itself to:

- changes in the configuration of the system controlled
- execution of different tasks

The field of application of the proposed controller is expected broad and applicable to a wide range of error spaces. But for direct evaluation, we test here the proposed approach using a simulated 3 DoF open-links robotics arms executing diverses tasks. Adaptability of the controller is tested as it is implemented in robots of different configurations. Periodic integration presented in the previous section is the only system used by the controller to adapt itself to the kinematic parameters of the robot as well as to the task being executed.

The simulation was created using Open Dynamic Engine [15] and the robotic arms moved in the 2D sagital plane subjected to gravity. The base of the robot is fixed on a ceiling and the starting position of the robots is $\theta_1 = 0$, $\theta_2 = 0$ and $\theta_3 = 0$, which corresponds to straight joints pointing downward. The sole inputs to the controller are the configuration of the robot (i.e. the joint angles θ_1, θ_2 and θ_3), the position of the end-effector in the cardinal space and λ, the angle between the last link and the vector defined by the end-effector and the target. No parameters are considered known, including that the system is an open-link system.

B. Results

1) Task 1: The first task consists in reaching the target following a straight line. The considered cartesian error-space is centered on the target and its basis is given by $[\boldsymbol{e_0}, \boldsymbol{e_1}]$; $\boldsymbol{e_0}$ is the vector defined by the target and the initial end-effector position; and $\boldsymbol{e_1}$ is a $\pi/2$ rotation of $\boldsymbol{e_0}$. The error vector position is the position $\boldsymbol{e} = [e_x, e_y]$ of the end-effector in the error-space. The task-specification function is given by $D(\boldsymbol{e}) = [-e_x, \frac{-e_y}{2}]$; which consists in reaching the target following the y abscisse of the error-space; i.e. the line defined by the target and the original position of the end-effector. We call this line the *reaching line*. Parameters are given in Table I. The task was executed on the 2DoF robot which joint characteristics are given in Table II. Results are given in Figure 6-a. On the right, the error components vs time is presented. On the left, the initial and final configurations of the robot are shown in dark grey. Lighter gray figures show intermediate configurations. The end-effector successfully moves toward the target following a straight line.

TABLE I
PARAMETERS OF THE CONTROLLER

PARAMETER	VALUE
k_p (physical system)	10
k_d (physical system)	10
k_p (virtual dynamic system)	1
k_d (virtual dynamic system)	0.1

TABLE II
PARAMETERS OF THE 2DOF ROBOT

PARAMETER	VALUE
joint 1	l=0.5[m], m=0.5[kg]
joint 2	l=0.4[m], m=0.4[kg]

TABLE III
PARAMETERS OF THE 3DOF ROBOT

PARAMETER	VALUE
joint 1	l=0.4[m], m=0.4[kg]
joint 2	l=0.3[m], m=0.3[kg]
joint 2	l=0.2[m], m=0.2[kg]

TABLE IV
PARAMETERS OF THE ALTERNATIVE 3DOF ROBOT

PARAMETER	VALUE
joint 1	l=0.3[m], m=0.3[kg]
joint 2	l=0.3[m], m=0.3[kg]
joint 2	l=0.3[m], m=0.3[kg]

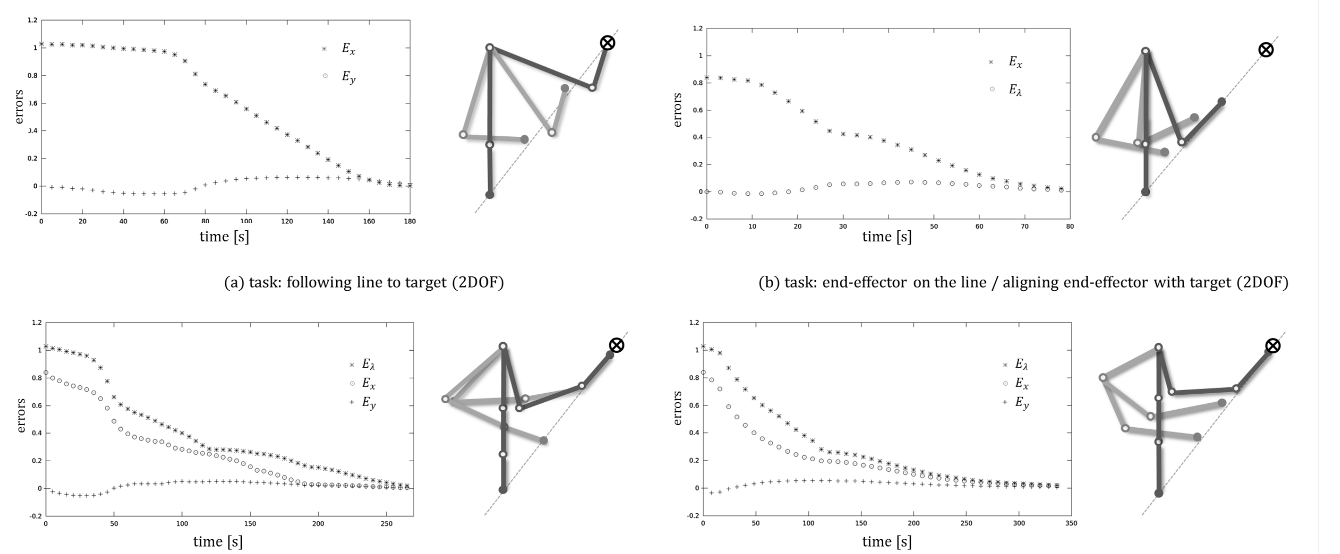
2) Task 2: The second task consists in aligning the end-effector on the target, while keeping the end-effector on the reaching line. The error-position vector is given by $[e_y, \lambda]$. The task-specification function is given by $D(\boldsymbol{e}) = [-e_x, -\lambda]$. The same 2 DoF robot as above was used and results are given in Figure 6-b. Note that the controller is completely identical to the one used for executing the first task, as only the error-space and the task definition have been changed. The end-effector successfully aligns toward the target while remaining close to the reaching line.

3) Task 3: The third task consists in the simultaneous exection of task 1 and 2, i.e. reaching the target following a straight line while aligning the last joint with the target. This task was executed by associating two sub-systems as shown on Figure 5. The resulting controller was tested using two different 3 DoF robots which joint characteristics are given in Table III and Table IV. Results shown in Figure 6-c and d show that the task is successfully performed.

V. DISCUSSION AND FUTURE WORK

The proposed controller showed adaptability as sole change in the inputs could have it accomplish two different simple tasks with success. Periodic integration resulted in the controller adapting itself to the configuration of the robot. The controllers did not include any prior knowledge of the task-space in which the errors were expressed; or any analytical model of the system in which they are embedded.

A limitation of this approach is that it is not suitable for control of underactuated systems where estimation of future reward is necessary. For such system, reinforcement learning might be preferable. In general, a theoretical analysis of the proposed approach is required to clearly define the



(a) task: following line to target (2DOF)
(b) task: end-effector on the line / aligning end-effector with target (2DOF)
(c) task: following line to target / aligning end-effector with target (3DOF)
(d) task: following line to target / aligning end-effector with target (3DOF, alt. configuration)

Fig. 6. Experimental results of control of simulated open-linked robotic arms performing different tasks

characteristics of the error-space necessary for application of the method. The stability analysis work of tacit-learning controllers, which started in [14], is also under progress.

This approach is applicable to abstract error spaces. While errors in a cartesian physical space is common in robotics, more abstract error space such as the one defined by markers used for characterizing the status of post-stroke patients can be considered. The other advantage of the proposed controller is its ability to continuously adapt to enviromental changes, e.g. recovery status of a post-stroke patient. Thus, we believe the proposed approach to be suitable for the adaptive control of robotic systems targeting efficient post-stroke rehabilitation. The embedded feature of continous adaptation is required as the patient capabilities change along its recovery status. Future work will focus on this application.

VI. CONCLUSION

We generalized the structure of the principle of tacit learning such as extending its applicability to a wide range of systems and error spaces. This was achieved by applying integration in a hierarchical fashion, in which the value of a virtual periodic dynamic variable is tuned for continuous adaptation. As learning is embedded in the controller, this later does not require any prior knowledge of the error-space in which the errors are expressed; or any analytical model of the system in which it is embedded. The advantage is that control and adaptation progress without explicit distinction between them. This approach is particular suitable for application in which features of the controlled system evolve with time.

REFERENCES

[1] S. Shimoda and H. Kimura, “Bio-mimetic approach to tacit learning based on compound control,” IEEE Trans. Syst., Man, Cybern.Part B, vol. 40, no. 1, pp. 7790, 2010.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323-9, 1986.

[3] Y. Kuniyoshi, Y. Yorozu, S. Suzuki, S. Sangawa, Y. Ohmura, K. Terada,and A. Nagakubo, “Emergence and development of embodied cognition: A constructivist approach using robots,” Progress in Brain Research, vol. 164, pp. 425445, 2007.

[4] R. S. Sutton and A. G. Barto, “Reinforcement learning,” MIT Press, 1998.

[5] K. Doya, “Reinforcement learning in continuous time and space,” Neural Computation, vol. 12, pp. 219245, 2000.

[6] Matsubara, J. Morimoto, J. Nakanishi, M. Sato, and K. Doya, “Learning CPG-based biped locomotion with a policy gradient method,” Robotics and Autonomous Systems, vol. 54, no. 11, pp. 911920, 2006.

[7] J. H. Holland, “Adaptation in natural and artificial systems,” MIT Press, 1992.

[8] M. Ito and M. Kano, “Long-lasting depression of parallel fiber-Purkinje cell transmission induced by conjunctive stimulation of parallel fibers and climbing fibers in the cerebellar cortex,” Neurosci. Lett., vol. 33, pp. 253258, 1982.

[9] M. F. Bear, B. W. Connors, and M. A. Paradios, “Neuroscience: Exploring the brain (third edition),” Lippincott Williams and Wilkinse, 2006.

[10] C. A. Janeway, P. Travers, M. Walport, and M. J. Shlomchik, “Immunobiology: The Immune System in Health and Disease - 5th Edition,” Garland Publishing, 2001.

[11] S. Shimoda and H .Kimura: “Neural Computation Scheme of Compound Control: Tacit Learning for Bipedal Locomotion,” SICE Journal of Control, Measurement, and System Integration, vol. 1, no. 4, pp.275283, 2008.

[12] S. Shimoda and H .Kimura: “Adaptability of tacit learning in bipedal locomotion,” IEEE Transactions on Autonomous Mental Development, Vol. 5, No. 2. pp. 152-161, 2013.

[13] M. Hayashibe and S. Shimoda, ”Synergetic Motor Control Paradigm for Optimizing Energy Efficiency of Multijoint Reaching via Tacit Learning”, Front. Comput. Neurosci., In Press

[14] Shingo Shimoda, Yuki Yoshihara, Kenji Fujimoto, Takashi Yamamoto, Iwao Maeda and Hidenori Kimura: “Stability analysis of tacit learning based on environmental signal accumulation” 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems October 7-12, 2012.

[15] Russell Smith:“Open Dynamics Engine” `http://www.ode.org/`