An Application of Receding-Horizon Neural Control in Humanoid Robotics

Serena Ivaldi, Marco Baglietto, Giorgio Metta, and Riccardo Zoppoli

Abstract. Optimal trajectory planning of a humanoid arm is addressed. The reference setup is the humanoid robot James [1]. The goal is to make the end effector reach a desired target or track it when it moves in the arm's workspace unpredictably. Physical constraints and setup capabilities prevent us to compute the optimal control online, so an off-line explicit control is required. Following previous studies [2], a receding-horizon method is proposed that consists in assigning the control function a fixed structure (e.g., a feedforward neural network) where a fixed number of parameters have to be tuned. More specifically a set of neural networks (corresponding to the control functions over a finite horizon) is optimized using the Extended Ritz Method. The expected value of a suitable cost is minimized with respect to the free parameters in the neural networks. Therefore, a nonlinear programming problem is addressed that can be solved by means of a stochastic gradient technique. The resulting approximate control functions are sub-optimal solutions, but (thanks to the well-established approximation properties of the neural networks) one can achieve any desired degree of accuracy [3]. Once the off-line finite-horizon problem is solved, only the first control function is retained in the on-line phase: at any sample time t, given the system's state and the target's position and velocity, the control action is generated with a very small computational effort.

Keywords: ERIM, robotics, receding horizon.

Serena Ivaldi and Giorgio Metta
Robotics, Brain and Cognitive Science Department, Italian Institute of Technology
e-mail: {serena.ivaldi,giorgio.metta}@iit.it

Serena Ivaldi, Marco Baglietto, Giorgio Metta, and Riccardo Zoppoli
Department of Communications, Computer and System Sciences,
University of Genoa, Italy
e-mail: {mbaglietto,rzop}@dist.unige.it

L. Magni et al. (Eds.): Nonlinear Model Predictive Control, LNCIS 384, pp. 541–550.
springerlink.com

1 Introduction

In robotics, the task of positioning the end effectors is fundamental: whenever a robot has to move its arm in order to grasp an object, track a moving target, avoid collision with the environment or just explore it, reaching is involved. Given the target position, estimated for example by a vision system, it is common practice to plan a suitable trajectory in the cartesian space and then to find the corresponding joint and torque commands. In industrial robotics, trajectories usually have a parametrized but fixed structure, e.g. splines or polynomials, or motor commands can be found analytically after the minimization of some Lyapunov function describing the reaching goal. In humanoid robotics, the focus is not only on reaching the target, but on how the target is reached, that is the criterion which a certain limb accomplishes while performing a movement or acting on the environment. One of the main goals of humanoid robotics is indeed to exploit redundancy and constraints of the humanoid shape to achieve behaviors that are approximately efficient as human movements. It is common belief that the human body moves "optimally" with respect to different cost functions, depending on action, limbs, task. In order to give a humanoid robot the chance to implement different motion criteria, it is necessary to provide a technique which allows finding optimal control commands for any given cost function. To this end, a Finite Horizon (FH) optimal control problem can be considered, but it is scarcely useful as generally the duration of the movements cannot be predicted *a priori*. Moreover, moving through a fixed horizon strategy could lead to a lack of responsiveness, whenever the target dynamics is too fast and no previous information is available to predict the target behavior. A Receding Horizon (RH) approach is suggested. Within the classical RH approach, at each time instant t, when the system state is $\mathbf{x}_t$, a FH optimal control problem is solved and a sequence of N optimal control actions is computed, $\mathbf{u}^{FH}_{0|t}, \mathbf{u}^{FH}_{1|t}, \ldots, \mathbf{u}^{FH}_{N-1|t}$ (corresponding to velocity, acceleration or torque commands, depending on the controller design), which minimize a suitable cost function affecting the motion performance; then only the first control vector is applied: $\mathbf{u}^{RH}_t = \mathbf{u}^{FH}_{0|t}$. This procedure is repeated at each instant t, thus yielding a feedback control law. Stabilizing properties of RH control have been shown for both linear and nonlinear systems, in continuous and discrete time, using the terminal equality constraints $\mathbf{x}_{t+N} = \mathbf{0}$ [4], relaxing it [5] and just imposing the attractiveness of the origin by means of penalty functions [2]. The classical RH technique assumes the control vectors to be generated after the solution of a nonlinear programming problem at each time instant: this assumption is generally unrealistic in the case of humanoid robotics, as the robot's and the target's dynamics are fast and the complexity of the problem increase with the number of DOF to control. In order to solve the optimization problem on-line, with the guarantee of respecting the temporal constraint, a suitable hardware and software are required, usually a real-time processing unit supporting fast and highly precise computations, directly connected to the robot' sensing and actuation devices, which is utterly complicated for complex kinematic structures. Unfortunately, different multi-level control architectures often do not support this control scheme. This is the case of our humanoid robot, James [1]. James is a humanoid torso, consisting of a head and a left arm , with the overall size of a 10 years old boy. Of the 7 DOF of the arm (3-shoulder, 1-elbow, 3-wrist), only 4 have been used in this paper (the wrist is considered as the end-effector), while numerical results are shown for the 2 DOF case. Torque is transmitted to the joints by rubber toothed belts, pulleys and stainless-steel tendons, actuated by rotary DC motors. The robot's motion can be controlled by sending position and velocity commands from a remote PC to 12 Digital Signal Processing (DSP) boards (Freescale DSP56F807, 80MHz, fixed point 16 bits), via CAN bus. DSP boards have limited memory and computation capability and cannot support more than simple operations, namely low level motor control (mostly PID controllers, 1KHz rate), signal acquisition and pre-filtering from the encoders. For this reason, implementing an on-line controller is impossible in the current setup: an explicit off-line RH controller is considered. The goal of this work is to design a feedback RH regulator for reaching tasks, with the requirement of being quick and reactive to changes, in particular to track a target moving unpredictably in the robot's workspace. We will also describe a technique which concentrates the computation of a time-invariant feedback optimal control law in an off-line phase , for every possible system and target states belonging to an opportune set of admissible states. The proposed algorithm consists of two steps. In the first step, a suitable sequence of neural networks is trained off line, so that they can approximate the optimal solutions of a stochastic FH control problem, which is generalized for every possible state configuration. In the second (online phase), only the first control law is applied, at each time instant. The Extended RItz Method (ERIM) [6] is chosen as a functional approximation technique. The use of feedforward neural networks (thanks to their well known approximation capabilities [7]) guarantees that the optimal solutions can be approximated at any desired degree of accuracy. We would like to remark that the computation demand is concentrated in the off-line phase, while in the on-line phase only the computation of a single control law is performed, thus yielding a fast response to unpredictable changes in the target's state, since we can do the computations quickly. The feasibility of this approach has already been tested on the control of a thrusts-actuated nonholonomic robot [8]. James can be modeled as an open kinematic chain. In the following we shall only focus on the arm motion control, in particular from the shoulder up to the wrist, which will be considered as the end effector of the kinematic chain, and neglect the rotation of the hand. Let us denote by $\mathbf{x}_c^r$ the cartesian coordinates of the end effector with respect to a base frame fixed to the robot waist, and by $\mathbf{q}$ the vector of the joint coordinates of the arm. Then the forward kinematics $\mathbf{x}_c^r = f_{\text{arm}}(\mathbf{q}),\ f_{\text{arm}} : \mathbb{R}^{n_q} \to \mathbb{R}^{n_c}$, can be easily found by measuring the length of the robot links and represent it with the Denavit-Hartenberg

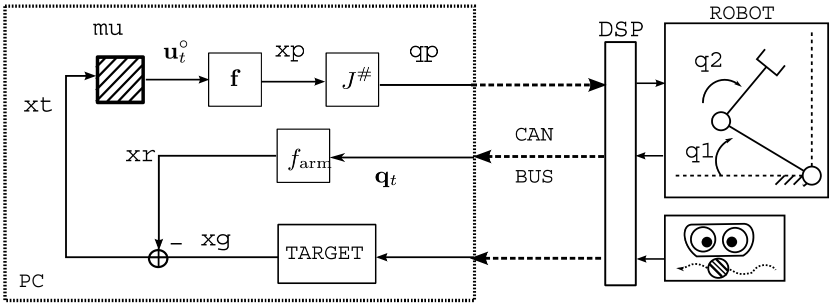

Fig. 1 James's arm control scheme. Velocity commands are sent through a CAN bus, while direct motor control is performed by DSP cards. The retrieving of the target's cartesian coordinates is not modeled, as it would require to discuss the robotic visual system. The arm kinematic model is reported. We indicated two arm joints (q_1, q_2), corresponding to the case of a 2 DOF arm $(n_q = 2)$

convention [9]. We shall denote by $\mathbf{x}_t^r$ and $\mathbf{x}_t^g$ the robot's end effector state vector and the target's one at time instant t. We remark that once the optimal control $\mathbf{u}_t^\circ$ is found, then the optimal velocity controls in the joint space can be easily computed with standard formulations, i.e., $\dot{\mathbf{q}}_t^\circ = J^\#(\mathbf{q}_t)\dot{\mathbf{x}}_t^{r\circ}$, where $J^\#$ denotes the Moore-Penrose pseudo-inverse of the jacobian matrix $J(\mathbf{q}) = \partial f_{\text{arm}}(\mathbf{q})/\partial\mathbf{q}$, being $\dot{\mathbf{x}}_c^r = J(\mathbf{q})\dot{\mathbf{q}}$. In particular, as explained in the previous section, they are computed by a standard Pentium based PC, then sent through the CAN bus to the DSP cards, where the low level control loop is performed. The control scheme is shown in Figure 1.

2 Receding Horizon Regulator: A Neural Approach

The goal of the reaching control problem is to find, at any time instant t, the optimal control $\mathbf{u}_t^\circ$ minimizing a suitable cost function, which is chosen so as to characterize the trajectories of the end effector reaching or tracking a target moving unpredictably in the robot workspace. We denote by $\mathbf{x}_t$, at time instant t, the difference between the end effector and the target cartesian coordinates and velocities ($\mathbf{x}_t \triangleq col(\mathbf{x}_t^g - \mathbf{x}_t^r, \dot{\mathbf{x}}_t^g - \dot{\mathbf{x}}_t^r)$). Let us represent the previous equations in the more general and compact form

$$\mathbf{x}_{t+1} = \mathbf{f}\left(\mathbf{x}_t, \mathbf{u}_t\right) \ , \ t = 0, 1, \ldots$$

where at the time instant t, $\mathbf{x}_t$ is the state vector, taking values from a finite set $X \subseteq \mathbb{R}^n$, and $\mathbf{u}_t$ is the control vector, constrained to take values from a finite set $U \subseteq \mathbb{R}^m$. At any time instant t, the desired state is $\mathbf{x}_t^* = 0$, meaning that the goal is to bring the difference between the end effector and the target to zero. By making this assumption, we implicitly apply a certainty equivalence principle: at time instant t, the target vector $\mathbf{x}_t^g$ is supposed to remain constant for N time instants, that is: $\mathbf{x}_{t+i+1}^g = \mathbf{x}_{t+i}^g, i = 0, \ldots, N-1$. We can now state a RH control problem.

Problem 1. At every time instant $t \geq 0$, find the RH optimal controls $\mathbf{u}_t^\circ \in U$, where $\mathbf{u}_t^\circ$ is the first vector of the control sequence $\mathbf{u}_{0|t}^\circ, \ldots, \mathbf{u}_{N-1|t}^\circ$ that minimize the FH cost functional

$$\mathcal{J}(\mathbf{x}_t) = \left\{ \sum_{i=0}^{N-1} h_i(\mathbf{x}_{t+i}, \mathbf{u}_{i|t}) + h_N(\mathbf{x}_{t+N}) \right\} .$$

The classical RH control assumes that at each time instant of control a FH control problem is solved, and a sequence of N optimal controls is found. As we previously discussed, this approach is not suitable in our case, for the hardware limitations imposed by the DSP cards. Therefore we will change the problem's formulation so as to be able to compute the control laws in an off-line phase.

Problem 2 (RH). For every time instant $t \geq 0$, find the RH optimal control law $\mathbf{u}_t^\circ = \mu_t^\circ(\mathbf{x}_t) \in U$, where μ_t° is the first control function of the sequence $\mu_{0|t}^\circ, \ldots, \mu_{N-1|t}^\circ$ that minimize the FH cost functional [1]

$$\bar{\mathcal{J}}_t = \mathop{E}_{\mathbf{x}_t \in X} \left\{ \sum_{i=0}^{N-1} h_i(\mathbf{x}_{t+i}, \mu_{i|t}(\mathbf{x}_{t+i})) + h_N(\mathbf{x}_{t+N}) \right\}.$$

Thanks to the time invariance of the system dynamics and of the cost function, $t = 0$ can be considered as a generic time instant. Then, a single (functional) FH optimization problem is addressed.

Problem 3 (FH). Find a sequence of optimal control functions $\mu_0^\circ, \ldots, \mu_{N-1}^\circ$, that minimize the cost functional

$$\bar{\mathcal{J}} = \mathop{E}_{\mathbf{x}_0 \in X} \left\{ \sum_{i=0}^{N-1} h_i(\mathbf{x}_i, \mu_i(\mathbf{x}_i)) + h_N(\mathbf{x}_N) \right\} \tag{1}$$

subject to the constraints $\mu_i^\circ \in \mathbf{U} \subseteq \mathbb{R}^m$ and $\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mu_i(\mathbf{x}_i))$.

The RH control strategy will correspond to use μ_0° as a time invariant control function, i.e., to apply $\mathbf{u}_t^{RH} = \mu^{RH}(\mathbf{x}_t) = \mu_0^\circ(\mathbf{x}_t)$.

[1] Hereinafter, the notation $\mathop{E}_{\xi}\{g(\xi)\}$ means the expectation of function g with respect to the stochastic variable ξ. It is important to notice that in Problem 1 the expectation is not necessary, because it is a deterministic problem.

2.1 From a Functional Optimization Problem to a Nonlinear Programming One

In order to solve Problem FH we shall apply the ERIM [6], by which the functional optimization problem is transformed into a nonlinear programming one. More specifically, we constrain the admissible control functions $\mu_0, \mu_1, \ldots, \mu_{N-1}$ to take on a fixed parametrized structure, in the form of one-hidden-layer (OHL) neural networks:

$$\hat{\mu}_i(\mathbf{x}_i, \omega_i) = col\left[\sum_{h=1}^{\nu} c_{hj}\varphi_h(\mathbf{x}_i, \kappa_h) + b_j\right] \tag{2}$$

where $\hat{\mu}_i(\cdot, \omega_i) : \mathbb{R}^n \times \mathbb{R}^{(n+1)\nu+(\nu+1)m} \mapsto \mathbb{R}^m$, $c_{hj}, b_j \in \mathbb{R}, \kappa_h \in \mathbb{R}^k, j = 1, \ldots, m$, being ν the number of *neurons* constituting the network. By substituting (2) into (1), calling ω_i the parameters of the i-th OHL network $\hat{\mu}_i(\mathbf{x}_i, \omega_i)$, the general functional cost $\bar{\mathcal{J}}(\mu_0, \mu_1, \ldots, \mu_{N-1})$ is turned into a function $\hat{\mathcal{J}}_\nu(\omega)$ which is only dependent on a finite number of real parameters, $\omega = col(\omega_i, i = 0, 1, \ldots, N-1)$. We can now restate Problem 3 as:

Problem 4 (FH_ν). *Find the optimal vectors of parameters $\omega_0^\circ, \ldots, \omega_{N-1}^\circ$ that minimize the cost function*

$$\hat{\mathcal{J}}_\nu = \underset{\mathbf{x}_0 \in X}{E}\left\{\sum_{i=0}^{N-1} h_i(\mathbf{x}_i, \hat{\mu}_i(\mathbf{x}_i, \omega_i)) + h_N(\mathbf{x}_N)\right\}$$

subject to the constraints $\hat{\mu}_i(\mathbf{x}_i, \omega_i) \in \mathbf{U} \subseteq \mathbb{R}^m$ and $\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \hat{\mu}_i(\mathbf{x}_i, \omega_i))$.

Then, for every time instant t, the time-invariant RH control law corresponds to $\mathbf{u}_t^{RH} = \hat{\mu}^{RH}(\mathbf{x}_t, \omega_0^\circ) = \hat{\mu}_0^\circ(\mathbf{x}_t, \omega_0^\circ)$.

2.2 Solution of the Nonlinear Programming Problem by Stochastic Gradient

The optimal parameters in the OHL control functions can be found by a usual gradient algorithm, i.e.

$$\omega_i(k+1) = \omega_i(k) - \alpha(k)\nabla_{\omega_i} \underset{\{\mathbf{x}_0\}}{E}\left\{\hat{\mathcal{J}}_\nu\left[\omega(k), \mathbf{x}_0\right]\right\},\ k = 0, 1, \ldots\ .$$

Within this context, it is impossible to calculate exactly all the gradient components, because of the stochastic nature of $\mathbf{x}_0$; then, instead of the gradient $\nabla_\omega E\left[\hat{\mathcal{J}}_\nu(\omega, \mathbf{x}_0)\right]$ a single "realization" $\nabla_\omega \hat{\mathcal{J}}_\nu(\omega, \mathbf{x}_0(k))$ is computed, where the stochastic variable $\mathbf{x}_0$ is generated randomly according to its known probability density function. Then a simple gradient steepest descent algorithm can be applied:

$$\omega_i(k+1) = \omega_i(k) - \alpha(k)\nabla_{\omega_i}\hat{\mathcal{J}}_\nu\left[\omega(k), \mathbf{x}_0(k)\right] + \eta(\omega_i(k) - \omega_i(k-1))$$

for $k = 0, 1, \ldots$, where we added a regularization term, weighted by $\eta \in [0,1]$, as it is usually done when training neural networks. The convergence of the method, which is known as *stochastic gradient*, is assured by a particular choice of the step size $\alpha(k)$, that must fulfill a set of conditions [10]. Of course, one has to compute the partial derivatives of the cost $\hat{\mathcal{J}}_\nu$ with respect to the parameters to be optimized, ω_i:

$$\frac{\partial \hat{\mathcal{J}}_\nu}{\partial \omega_i} = \frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{u}_i}\frac{\partial \hat{\mu}_i(\mathbf{x}_i, \omega_i)}{\partial \omega_i} \; .$$

The proposed algorithm for the computation of the optimal parameters consists in two phases, a forward and a backward one, and in a back-propagation technique. In the *forward phase* we "unroll" the system and the neural controllers in time, making the feedback explicit. At iteration step k, given the initial state $\mathbf{x}_0$, we compute all the state and controls generated by the sequence of OHL networks that is $\mathbf{u}_i = \hat{\mu}_i(\mathbf{x}_i, \omega_i(k))$, given $\mathbf{x}_0, \mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{u}_{i-1})$, $i = 1, \ldots, N$. Then we can compute all the partial costs $h_i(\mathbf{x}_i)$, $h_N(\mathbf{x}_N)$. In the *backward phase*, we compute all the gradient components and "back-propagate" them through the networks' chain. The recursive propagation is described by the following equations, for $i = N-1, N-2, \ldots, 0$:

$$\begin{aligned}
\frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{u}_i} &= \frac{\partial h_i(\mathbf{x}_i, \mathbf{u}_i)}{\partial \mathbf{u}_i} + \frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{x}_{i+1}}\frac{\partial f(\mathbf{x}_i, \mathbf{u}_i)}{\partial \mathbf{u}_i} \\
\frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{x}_i} &= \frac{\partial h_i(\mathbf{x}_i, \mathbf{u}_i)}{\partial \mathbf{x}_i} + \frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{x}_{i+1}}\frac{\partial f(\mathbf{x}_i, \mathbf{u}_i)}{\partial \mathbf{x}_i} + \frac{\partial \hat{\mathcal{J}}_\nu}{\partial \mathbf{u}_i}\frac{\partial \hat{\mu}_i(\mathbf{x}_i, \omega_i)}{\partial \mathbf{x}_i}
\end{aligned}$$

initialized by $\partial \hat{\mathcal{J}}_\nu / \partial \mathbf{x}_N = \partial h_N(\mathbf{x}_N)/\partial \mathbf{x}_N$.

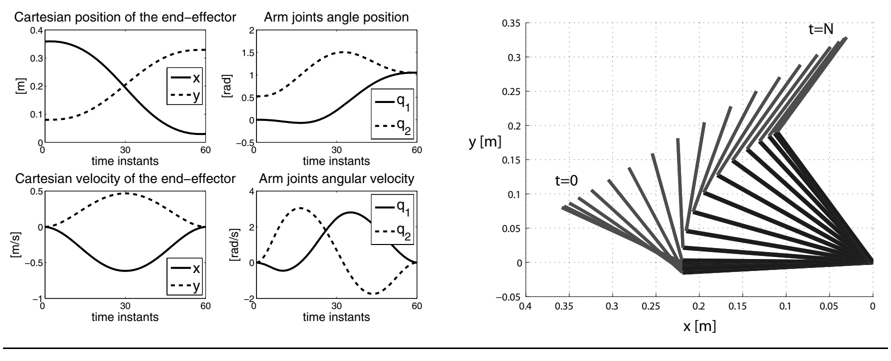

Fig. 2 A minimum jerk movement of James'arm: cartesian and joints position and velocity are shown, as well as samples of the planar trajectory. The neural approximation and the analytical solution [11] coincide (m.s.e. $\cong 10^{-7}$)

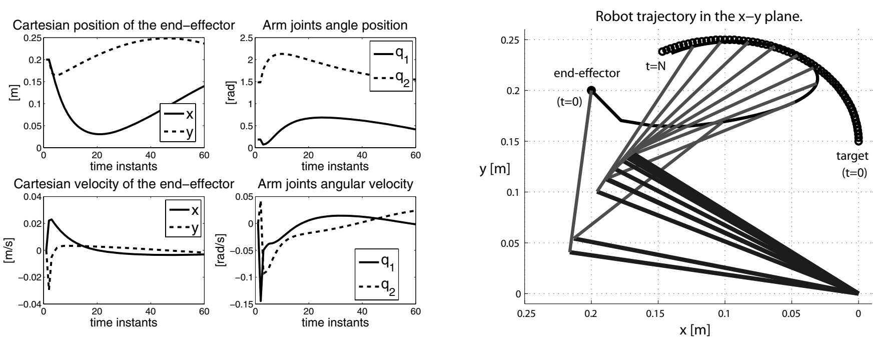

Fig. 3 James' left end effector tracking a target moving in an unpredictable way, according to cost function (3), where $V_i = diag(1.0, 80.0, 5.0, 10.0)$, $V_N = 40I$. Moreover, $N = 30, \nu = 40$

3 Results

Many neuro-computational studies investigate the arm motion on a plane, considering the arm as a two-rotative joints limb. In this case, it has been shown that the human arm movement can be approximated by the function optimizing the following cost function (*minimum jerk* principle) [11]:

$$\mathcal{J} = \int_0^T \left[\left(\frac{d^3 x^r}{dt^3} \right)^2 + \left(\frac{d^3 y^r}{dt^3} \right)^2 \right] dt \ .$$

This criterion has been chosen to verify the effectiveness of the proposed method. We set $n_q = n_c = 2$ to consider James' arm as a two-link rigid body, moving on a planar surface, $T = 60$, $\nu = 40$, and used approximatively 10^9 samples for the off-line training of the neural networks. Results are shown in Figure 2. The method has been also tested with a different cost function:

$$\mathcal{J} = \sum_{i=t}^{t+N-1} \mathbf{c}(\mathbf{u}_i) + \mathbf{x}_{i+1}^T V_{i+1} \mathbf{x}_{i+1} \tag{3}$$

where the criterion for the task accomplishment is a tradeoff between the minimization of the energy consumption (for physical limits, it is important not to exceed in the maximum rated current consumption) and the "best" end-effector proximity to the target during and at the end of the manoeuvre (it could not be able to reach it perfectly, as a consequence of the unpredictable behavior of the target or the robot's intrinsic physical limits). Weight matrices V_i are chosen such as to obtain reasonable compromise between the attractiveness of the target and the energy consumption, whereas $c(u_t^j), j = x, y$ is a nonlinear but convex function (the same reported in [8]). An example of a

RH trajectory during a tracking task are shown in Figure 3. We remark that the constraints on the admissible values of $\mathbf{x}_t$ and $\mathbf{u}_t$ are always verified. To be more precise, the classical OHL networks were slightly modified, specifically by adding two bounded sigmoidal functions $\sigma(z) = U \tanh(z)$ to the final output layer: with this choice, the constraints on the control values can be removed from the problem formulation since the neural networks already embed them.

4 Conclusion

This paper focused on the computation of a neural time invariant feedback control law optimized off-line. The on-line computation of the control action is efficient, as it consists only of few mathematical operations. We point out that the requirement of computing control values in real-time as fast as possible is strict. Given that this method has been designed to be applied to a full body humanoid robot, we concentrated in making the computation of the control law as efficient as possible. We have presented simulations to clarify the problem. Early experiments on James, controlling 2 DOF, have confirmed the effectiveness of the proposed approach. Simulations for the control of the 4 DOF arm are currently ongoing. In the future, the control scheme will take into account singularities, redundancies of the kinematic chain, and delays which have been neglected for the moment.

Acknowledgement. This work is supported by the RobotCub project (IST-2004-004370), funded by the European Commission through the Unit E5 "Cognitive Systems, Robotics and Interfaces".

References

1. Jamone, L., Metta, G., Nori, F., Sandini, G.: James: a humanoid robot acting over an unstructured world. In: Int. Conf. on Humanoids Robots, Italy (2006)
2. Parisini, T., Zoppoli, R.: A receding horizon regulator for nonlinear systems and a neural approximation. Automatica 31, 1443–1451 (1995)
3. Kurkova, V., Sanguineti, M.: Error estimates for approximate optimization by the Extended Ritz method. SIAM J. on Optimization 15, 461–487 (2005)
4. Keerthi, S., Gilbert, E.: Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations. Journ. of Optim. Theory and Applications 57, 265–293 (1988)
5. Michalska, H., Mayne, D.: Robust receding horizon control of constrained nonlinear systems. IEEE Trans. on Automatic Control 38, 1623–1633 (1993)
6. Zoppoli, R., Sanguineti, M., Parisini, T.: Approximating networks and Extended Ritz Method for the solution of functional optimization problem. Journ. of Optim. Theory and Applications 112, 403–439 (2002)
7. Barron, A.: Universal approximation bounds for superposition of a sigmoidal function. IEEE Trans. on Information Theory 39, 930–945 (1993)

8. Ivaldi, S., Baglietto, M., Zoppoli, R.: Finite and receding horizon regulation of a space robot. In: Int. Conf. on Mathem. Probl. in Engin., Aerospace and Sciences, Genoa, Italy (2008)
9. Sciavicco, L., Siciliano, B.: Modeling and Control of Robot Manipulators, 2nd edn. Springer, London (2000)
10. Kushner, H.J., Yang, J.: Stochastic approximation with averaging and feedback: rapidly convergent "on-line" algorithms. IEEE Trans. on Automatic Control 40, 24–34 (1995)
11. Richardson, M.J., Flash, T.: On the Emulation of Natural Movements by Humanoid Robots. In: Int. Conf. on Humanoids Robots, Boston, USA (2000)