Neural adapted controller learned on-line in real-time

Adam Piłat*, Andrzej Turnau**

** AGH – University of Science and Technology, Department of Automatics, Al. Mickiewicza 30, 30-059 Kraków, Poland (e-mail: ap@agh.edu.pl).*
*** AGH – University of Science and Technology, Department of Automatics, Al. Mickiewicza 30, 30-059 Kraków, Poland (e-mail: atu@agh.edu.pl).*

Abstract: An adapted controller consisting of a PD structure aided by a neural network is considered. It is dedicated to an unstable magnetic levitation system (MLS) to maintain a levitated sphere in the real-time. The main motivation is to create a self-tuning intelligent device that can easily change the controller parameter estimates based on adaptive update rules. How to formulate these rules in order to not disturb or ruin the system stability is the most serious question. This issue is discussed and verified by real-time experiments carried out at our apparatus that enables to vary its parameters. The neural network learned on-line guarantees the robustness of the control despite the system parameter are changed and disturbances are introduced. One can expect a better stabilizing performance as well.

1. INTRODUCTION

A special MLS apparatus has been constructed to perform a stabilization experiments in the real-time and illustrate the declared intelligent behaviour of the controlled device (Piłat, Turnau 2005). The MLS seems to be a right choice. This is the frictionless unstable system with a simple but not trivial dynamic. Hence it is an incessant temptation to develop the intelligent control technology based on the MLS apparatus. The paper starts from a short description of the experimental apparatus and its model. Next, the PD and NN adaptive control strategy working with sampling frequency 2kHz in the real-time is presented. The adaptive NN tuning algorithm and stability issues are discussed. Two control experiments performed in the real-time are presented. One can notice how the NN controller weights and biases are tuned and how accurate is the performance of the combined PD + NN controller. Conclusions are added.

2. MAGNETIC LEVITATION LABORATORY SET-UP AND ITS MODEL.

MLS consists of: the electro-magnet, the suspended hollow steel sphere, the sphere position sensors, computer interface board and drivers, a signal conditioning unit, connecting cables. MLS is a nonlinear, open-loop unstable and time varying dynamical system. The basic principle of MLS operation is to apply the voltage to an electromagnet to keep a ferromagnetic object levitated. The object position is determined through a sensor. Additionally the coil current is measured to explore identification and multi loop or nonlinear control strategies. To levitate the sphere a real-time controller is required. The equilibrium stage of two forces (the gravitational and electro-magnetic) has to be maintained by this controller to keep the sphere in a desired distance from the magnet. The MLS laboratory set-up is equipped with the current driver that cooperate with electromagnet. It results in damp-less system without a phase shift within the range from 0 to 10 Hz. Due to the current driver it was possible to simplify the MLS construction and diminish a time constant of the actuator relatively to an MLS supplied by a voltage PWM signal. The details of this new MLS current driver will appear in (MLSEMi 2008). Let us consider a model of the MLS steered by a coil current as the control variable (see Piłat 2009). The model dynamics is described by the nonlinear differential equations. This equation can be linearized at the fixed state variable values.

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -K_{em} i^2 x_1^{-2} m^{-1} + g + F_{Ext} m^{-1} \end{aligned} \tag{1}$$

where: x_1 (measured in [m]) and x_2 (measured in [m/s]) denote the sphere distance from the electromagnet and the sphere velocity, m is the sphere mass equal to 0.056 [kg], g is the gravity equal to 9.81 m/s2, K_{em} is the construction constant that characterizes the electromagnet equal to 2.913·10-5 [Nm^2/A^2], i (measured in [A]) denotes the current of the electromagnet coil and F_{Ext} denotes the external force generated by the lower electromagnet in the same direction as the gravity force. The distance and current satisfy: $x_1 \in (0,0.02]$ [m]. The linear model is derived under the assumption $F_{\mathrm{Ext}} = 0$.

$$\dot{x} = Ax + Bi\ ,\ A = \begin{bmatrix} 0 & 1 \\ \alpha & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ \beta \end{bmatrix} \tag{2}$$

where:

$\alpha = 2K_{em} i_0^2 x_{10}^{-3} m^{-1}$ [s^{-2}] and $\beta = -2K_{em} i_0 x_{10}^{-2} m^{-1}$ [mA^{-1}s^{-2}].

The linearized model at $(x_{10},\ 0)$ and $i_0 = x_{10}\sqrt{mgK_{em}^{-1}}$ together with the PD feedback can be in several modes of operation. The feedback parameters: i_0, K_p (proportional gain) and K_d (derivative gain) determine the model behaviour. The feedback signal is expressed by the formula

$$i_{PD} = i_0 + K_p e + K_d \dot{e} \tag{3}$$

where $e = x_{10} - x_1$. This signal is modified by the NN controller

$$i = \alpha_u i_{PD} + (1 - \alpha_u) i_{NN} \tag{4}$$

3. PD AND NN ADAPTIVE CONTROL ISSUES.

To perform real control experiments the following controller built in Simulink is used (see Fig. 1). The *ML I/O* block represents the drivers that communicate with the real MLS laboratory set-up. One can notice: *PD Controller* and *NN S-function*, the aggregated *PD/NN Controller* steered by the coefficient α_u, the *External Excitation* block (applied to control the lower electromagnet), the *Desired Position [m]* block and *Scope*. It is not necessary to explain how all these blocks work except *NN S-function*. It is not an ordinary NN but one equipped with the learning mechanism.

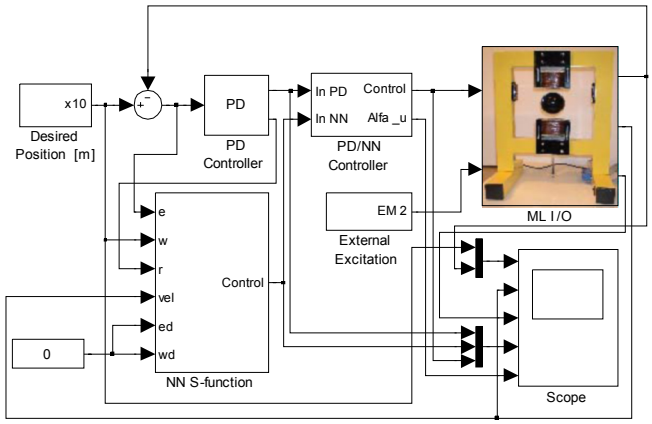

Fig. 1. PD and NN adaptive controller applied to the real-time MLS experiments

The *NN S-function* block has the structure shown in Fig. 2. The NN adopted convention is from the books (Lewis et al 2002) and (Lewis et al 2004). There are the hidden-layer with the weights vector *V* and the output-layer with the weights vector *W*. The NN realizes at its output the following function

$$u_{NN} = W^T \sigma(V^T x) \tag{5}$$

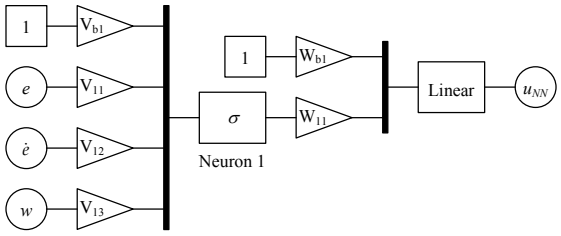

Fig. 2. The internal structure of the NN controller

The nonlinear sigmoidal activation function has the form:

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}}, \text{ where } z_i = \sum_{j=1}^{J} V_{ij}^T \cdot x_j \ . \tag{6}$$

The *i* index denotes a neuron number in the hidden-layer. The *j* index denotes a coordinate of the input vector *x* (*J* denotes the *x* vector dimension). The input vector

$$x = [1 \quad e \quad \dot{e} \quad w]^T \tag{7}$$

is augmented by a "1" – the bias is added. Because our NN consists of one neuron in the hidden-layer and posses the linear activation function in the output-layer the u_{NN} output is calculated on the basis of the following simple formula:

$$u_{NN} = W_{b1} + W_{11}\sigma\left([V_{b1}\ V_{11}\ V_{12}\ V_{13}][1\ e\ \dot{e}\ w]^T\right) = W_{b1} + W_{11}\sigma\left(V_{b1} + \sum_{i=1}^{3} V_{1i}\right) \tag{8}$$

As far as the NN learning procedure is concerned the following facts are important. The learning begins at once when the experiment starts. The *V* weights and biases of the hidden- layer are initialised randomly but the *W* weight and bias of the output-layer are set to zero. Thus the NN at the beginning of the learning procedure is not present. Then the NN learns by modifying the weights. The weights are updated in every sampling period. The following steps are performed. The output of the hidden-layer is calculated. The weights of the hidden-layer are updated. The weights of the output-layer are updated. The NN output is calculated. To update the weights the gradient algorithm based on the back-propagation of the error is used. It is obvious from formula (6) that the output of the hidden-layer is a nonlinear in the weights *V*. We can ask how to construct weight tuning algorithms yielding stability of the closed-loop system. The hidden-layer output error $\tilde{\sigma}$ is expressed as:

$$\tilde{\sigma} = \sigma(V^T x) - \sigma(\hat{V}^T x) \tag{9}$$

where: $\hat{V}^T$ denotes the hidden weights estimates vector. Equation (9) can be expand to the Taylor series:

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{10}$$

where: σ' denotes the first derivative of activation function that can be easily computed, $\tilde{V}^T$ denotes the hidden weights error vector and $O(\tilde{V}^T x)^2$ represents higher order terms that can be neglected. Finally, the last equation can be rewritten in the form

$$\tilde{\sigma} = \hat{\sigma}'\ \tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{11}$$

The hidden output error $\tilde{\sigma}$ nonlinear in the hidden weight error $\tilde{V}$ (see formula (9)) is re-placed by $\tilde{\sigma}$ linear in $\tilde{V}$ plus the higher order terms (see formula (11)). The last formula can be combined with the output-layer weights *W*. This layer has a linear activation function. Therefore, we obtain a simple tuning laws for $\hat{V}$ and $\hat{W}$ that guarantee closed-loop stability.

4. STABILITY OF THE CLOSED LOOP ERROR DYNAMICS.

In this section we focused our attention on the static feedforward NN that is turned into a dynamic NN when a feedback loop around it is closed. In fact, we have an outer PD tracking loop (3) and an inner linearization NN loop. Both loops operations are summed in formula (4). We follow here deliberations and proofs presented in two books: (Lewis et al. 2002) and (Lewis et al. 2004). The new intelligent NN control approach to control robot manipulators is considered.

The authors provide a very interesting NN learning methodology (no preliminary learning phase is required). NN weight on-line tuning algorithms are based on a backpropagation equipped with a special k term called *e*-modification (see Narendra, Annaswamy 1987). As far as the closed loop control dynamics stability issues are concerned it helps a lot to find out such a Lapunov function L and establish that $\dot{L}$ is negative outside a compact set in the tracking and weight vector norms plane. This condition is necessary to proof that both vector norms are UUB *ultimately uniform bounded*.

To be consistent with the notation used in the mentioned books and to not interfere with the NN input vector x (7) we will use from now the variable q that denotes the variable x_1 in formula (1). Instead of (1) we obtain the second order differential equation describing the MLS dynamics

$$q^2\ddot{q} - q^2 g = -K_{em} i^2 m^{-1} \tag{12}$$

If it is defined a desired sphere trajectory $q_d(t) \in \mathbf{R}^2$ then the tracking and filtered tracking errors are

$$e(t) = q_d(t) - q(t) \tag{13}$$

$$r = \dot{e} + \Lambda e \tag{14}$$

where Λ is a symmetric positive definite design parameter matrix. Important is to find a controller in such a way that $r(t)$ is bounded, because $e(t)$ becomes also bounded. Differentiating $r(t)$ we obtain

$$\dot{r} = \ddot{e} + \Lambda\dot{e} = \ddot{q}_d - \ddot{q} + \Lambda\dot{q}_d - \Lambda\dot{q} \tag{15}$$

Hence

$$\ddot{q} = -\dot{r} - \Lambda\dot{q} + \ddot{q}_d + \Lambda\dot{q}_d \tag{16}$$

If q_d is constant then two last terms in (16) are equal zero – the sphere is stabilized at a constant level. If $\ddot{q}$ from (16) is inserted to (12) and instead the right hand side of (12) we put $\tau = -K_{em} i^2 m^{-1}$ then we obtain

$$q^2\dot{r} = -\tau + q^2\ddot{q}_d + q^2\Lambda\dot{q}_d - q^2\Lambda\dot{q} - q^2 g \tag{17}$$

The last equation can be rewritten in the form

$$q^2\dot{r} = -\tau + f \tag{18}$$

where the nonlinear MLS function that will be approximated by the NN is

$$f(x) = q^2(\ddot{q}_d + \Lambda\dot{e}) - q^2 g \tag{19}$$

In fact, the components of the vector x shown by (7) may be enlarged to the following set $x \equiv [e^T \ \dot{e}^T \ q_d^T \ \dot{q}_d^T \ \ddot{q}_d^T]^T$. The most important fact is that a required input control τ consists of three components

$$\tau = \hat{f} + K_v r - v \tag{20}$$

with $K_v = K_v^T > 0$ a gain matrix, and $\hat{f}(x)$ an estimate of the MLS function $f(x)$. The signal v is added to compensate disturbances. If the control (20) is used then the closed-loop error dynamics has the form

$$q^2\dot{r} = -K_v r + \tilde{f} + v \tag{21}$$

where $\tilde{f} = f - \hat{f}$ is the function error. It is assumed that the desired trajectory is bounded

$$\left\| \begin{matrix} q_d(t) \\ \dot{q}_d(t) \\ \ddot{q}_d(t) \end{matrix} \right\| \le q_B \tag{22}$$

q_B is known positive number. It is shown in Lewis et al. 2002 that

$$\|x\| \le q_B + c_0\|r(0)\| + c_2\|r\| \tag{23}$$

If we use the NN to approximate $\hat{f}$ then we apply (5) with the weight estimates $\hat{W}$ and $\hat{V}$. Due to the NN approximation property there always exists a NN that the estimate $\hat{f}$ is accomplished with a given accuracy ε. To apply the NN we have to define an initial set of the tracking errors and an initial set of the initial weights. It is tightly connected with a tune manner of the NN weights. The closed-loop stability has to be guaranteed. The approximation accuracy ε of the NN determines the set of initial tracking errors

$$S_r = \left\{ r \mid \|r\| < \frac{b_x - q_B}{c_0 + c_2} \right\} \tag{24}$$

where $b_x > q_B$ is the ball radius for all x satisfying the NN approximation property. As far as weights are concerned one must assume that they are constant and bounded

$$\|W\|_F \le W_B \tag{25}$$

Because the NN is used to approximate function (19) the required input control τ has the form

$$\tau = \hat{W}^T\sigma(\hat{V}^T x) + K_v r - v \tag{26}$$

Consequently, the closed-loop filtered error dynamics (21) has the form

$$q^2\dot{r} = -K_v r + \tilde{W}^T\hat{\sigma}'\tilde{V}^T x + v + \varepsilon \tag{27}$$

Finally, we refer to two theorems 8.4-1 and 8.5-1 given in Lewis et al. 2004. Our notation given in the formulas is compatible to the book notation. Therefore the formulas can be directly applied. Beside the estimations shown in the theorems one can notice the Lapunov function candidates and its derivatives. The theorems guarantees stability of the filtered error closed-loop dynamics. The proposed NN adaptive technique is a significant improvement over other NN approaches. We do not need to define initial stabilizing weights. Such a requirement common to other NN techniques is very restrictive to be applied.

5. REAL-TIME CONTROL EXPERIMENTS.

5.1 Experiment 1.

In the first experiment at the beginning a pure PD control is applied. Afterwards the NN controller turns on and little by little increases its operation. Its contribution to the entire control is steered by the 1-α_u expression value. Fig. 3 shows how the PD control contribution rate α_u diminishes in time. Concurrently, the NN control contribution rate 1-α_u enlarges.

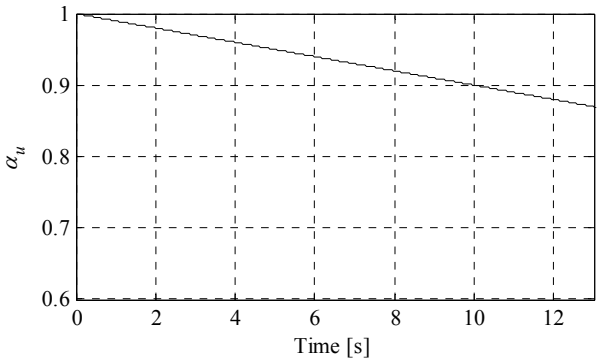

Fig. 3. PD contribution to control rate vs. time

One can notice how the weights and biases are updated during learning. The hidden-layer weights and bias learned on-line are illustrated in Fig. 4.

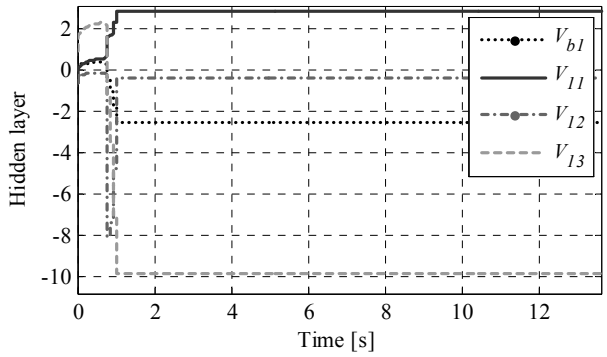

Fig. 4. The hidden-layer weights and bias learned on-line

Respectively Fig. 5. presents one weight and one bias of the output-layer. The bias significantly changes its value while the NN is learned. It results in the static error cancellation overcome by the trained NN.

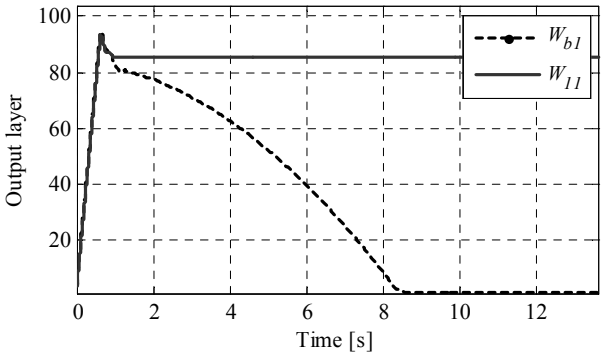

Fig. 5. The output-layer weight and bias learned on-line

An interesting behaviour of the levitating sphere is observed around the eighth second. The sphere get closer to the desired value level. Finally, the position error is reduced to zero (see Fig. 6).

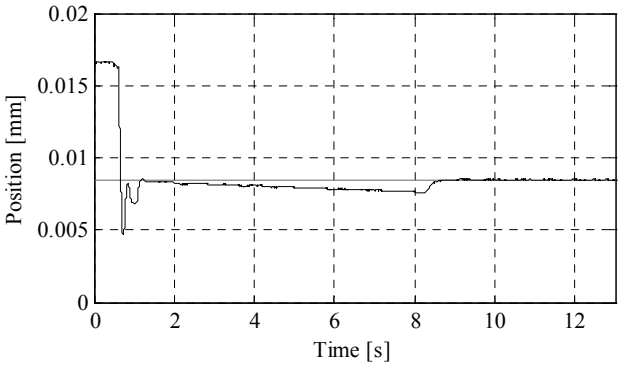

Fig. 6. The sphere distance from the electromagnet vs. time

Hereafter, the PD control influence is significantly reduced. The NN control takes charge. Of course, the PD controller still operates aided by the NN controller (see Fig. 7). This cooperation seems to be the most desirable approach to the control. It is a well known fact among control engineers that a simple PD structure does not guarantee successful stabilization of MLS under its parameters variation and an influence of disturbances. A promising method is to built an extra NN adaptation loop equipped with an on-line learning mechanism. One can notice an abrupt change at the NN output. The dormant NN is awaken. The aggregated control is shown in Fig. 8.

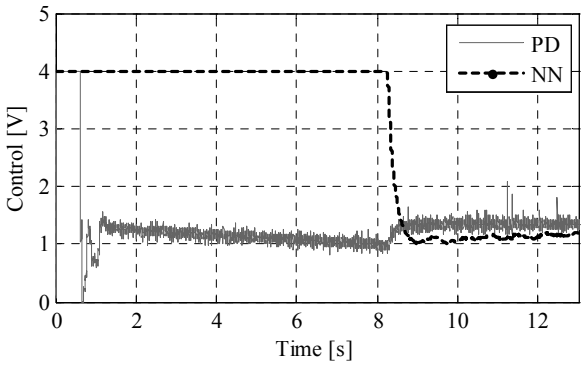

Fig. 7. PD and NN controls shown separately

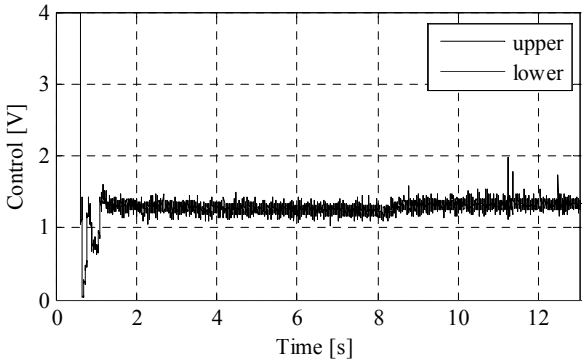

Fig. 8. Aggregated PD and NN controls

The goal of this experiment is to illustrate how the NN little by little takes charge and contributes to the entire control

loop. The NN impact is little by little included to the entire control. A steady state control error is reduced to zero. That means that the NN has been sufficiently trained to become a proper control term in the entire closed loop system.

5.2 Experiment 2.

The aim of the second experiment is to show how robust is our adaptive controller described in the first experiment. We expect it would operate properly despite parameters changes and disturbances. To demonstrate the case one can replace the levitating sphere to a new one different in mass.

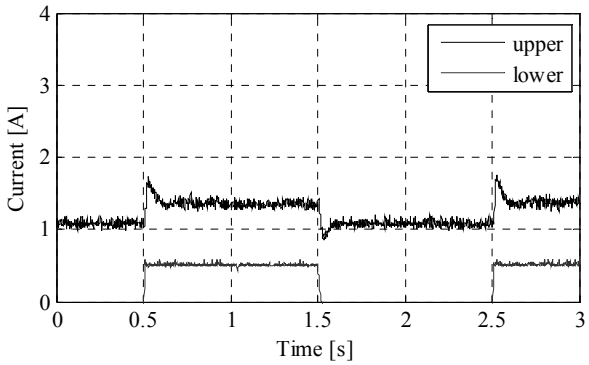

Fig. 9. Current at the lower electromagnet (it is turned on and off) Current response at the upper electromagnet

If MLS adopts itself to this abrupt change of mass it will be recognized as the robust. To avoid a manual spheres exchange we can proceed in a different way convenient to be automated. We use the second lower electromagnet of the apparatus to generate an extra magnetic field (see the square wave current signal in Fig. 9). It resembles in effect the masses replacement phenomena. The adaptive current control signal (PD + NN) of the upper electromagnet is shown in the same figure. How the PD control contribution rate α_u diminishes in time in this experiment is illustrated in Fig. 10.

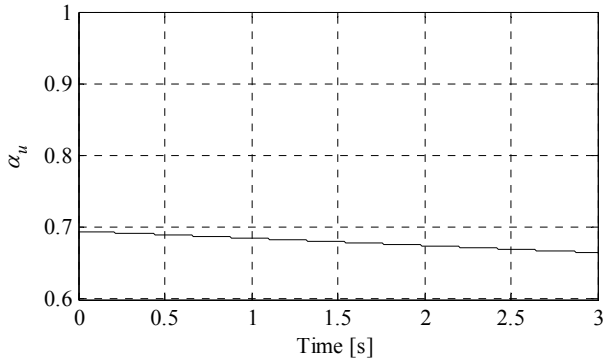

Fig. 10. PD contribution to control rate vs. time

Again it is interesting to show how behave the weights and biases during learning. In Fig. 11 the hidden-layer weights are illustrated.

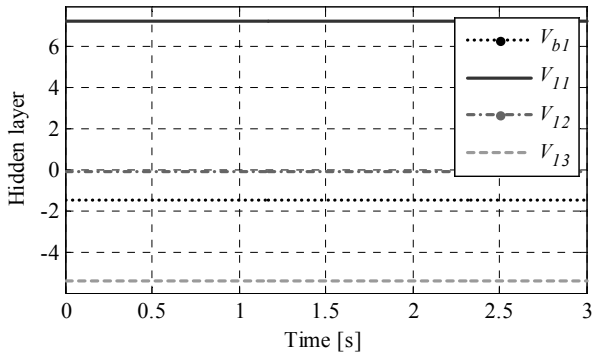

Fig. 11. The hidden-layer weights learned on-line

None sudden change in values is observed. Simply the NN has been trained. We do not watch the beginning of the learning procedure as it was illustrated in the first experiment. However, the bias in the output-layer (see Fig. 12) is accurately following the variations of the current in the lower electromagnet (notice this signal in Fig. 9). The separate behaviours of both controls are illustrated in Fig. 13. This experiment of the NN learned on-line is quite interesting in virtue of a periodical repetitive adaptive control applied to the real-time MLS. The NN is permanently trained. It reacts at once to a large parameter change. Fig. 14. presents the sphere distance to the upper electromagnet measured in millimetres. These experiments confirm that the proposed controller operates in a stable way. Due to the adaptation neural mechanism the sensitivity to the system parameter changes is significantly reduced.

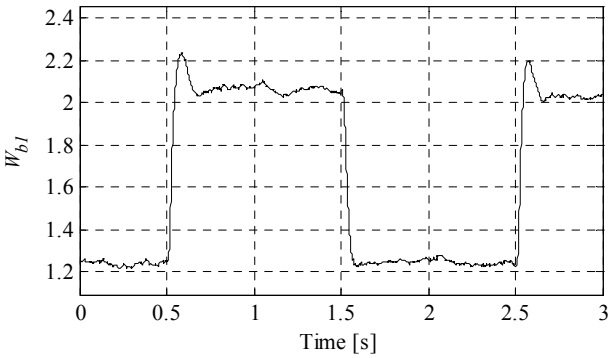

Fig. 12. The output-layer bias learned on-line

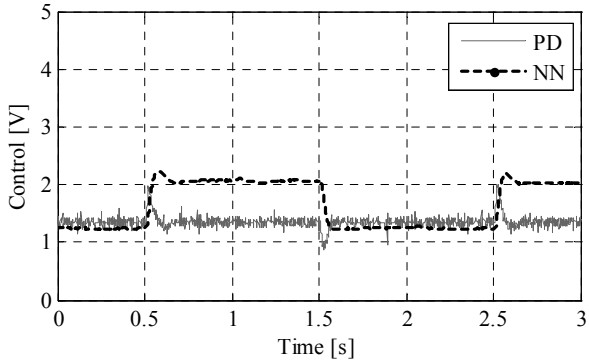

Fig. 13. PD and NN control responses shown separately

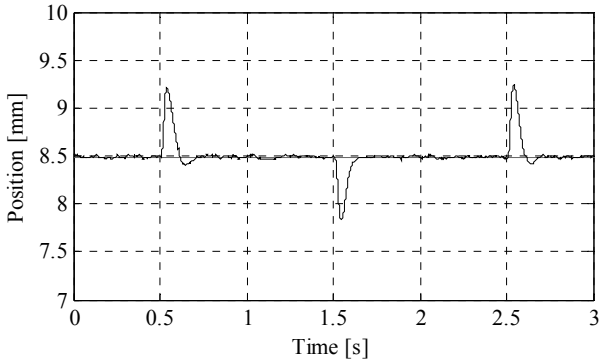

Fig. 14. Sphere distance from the upper electromagnet in millimetres

6. CONCLUSIONS

The learning algorithm built into the NN learning code guarantees the closed loop control operation in stable ranges. It is obvious that a PD controller can not reduce the steady state error consequent upon a new level of the levitating sphere. The error elimination becomes realistic if the controller structure is supplemented by the learned on-line NN. The very important control issue is that the NN is the nonlinear system. Therefore it is a proper approximator of MLS structural nonlinearities. The PD equipped with the NN learned on-line becomes a flexible robust controller.

Acknowledgement

This research was supported by the AGH grant 11.11.120.768

REFERENCES

Lewis F. L., Campos J. Selmic R. (2002), Neuro-fuzzy control of industrial systems with actuator nonlinearities, SIAM, Philadelphia.

Lewis F. L., Dawson D. M., Chaouki T. A. (2004), Robot manipulator control. Theory and practice. Marcel Dekker, Inc. New York.

MLS2Emi (2008), Magnetic Levitation System Users Guide, INTECO Ltd. Kraków.

Narendra K. S., Annaswamy A. M. (1987), A new adaptive law for robust adaptive control without persistent excitation, IEEE Trans. Automat. Control, vol. 32, pp. 134-145, Feb.

Piłat A., Turnau A. (2005), Self-organizing fuzzy controller for magnetic levitation system, Computer Methods and Systems, Cracow, Poland 14-16 November

Piłat A. (2009), Stiffness and damping analysis for pole placement method applied to active magnetic suspension. AGH Automatics, vol.1 to be published