Neural Networks for Synthesizing Linear Feedback Control Systems Via Pole Assignment

Marko Kovandžić, Stanoja Bunuševca 20/82, 18000 Niš, Srbija

marko.kovandzic@utva.rs, marko.kovandzic@gmail.com

Abstract

Artificial neural networks involve a network of simple processing elements (artificial neurons) which can exhibit complex global behavior. This behvavior is determined by the connections between the processing elements and element parameters. Artificial neural networks can be easily trained to perform a particular function by adjusting the values of the connections (weights) between elements. In this manner, neural networks provide sophisticated and efficient information processing which makes them applicable for solving broad range of artificial intelligence problems. Becouse of ability to solve problems that are difficult for solving for conventional computers or human beings (speech recognitionn, image analysis, classification, adaptive control, autonomus robots control...) neural networks have been applied in almost all branches of engenering systems. One of the applications is in control systems where artificial recurrent neural networks provide an effective on-line dynamic approach for synthesizing linear control systems via pole assignment. To ensure control system stability recurent neural network performs self tuning of control parameters as respond to system parameters changing. This kind of control system is called adaptive control system. If the state variables measuremant is not possible or it is very slow or expensive neural network can be used for system identification, in other words, for determination of unknown state variables.

Keywords

Linear feedback control, pole assignment, recurrent neural network, multilayer neural network, adaptive systems, system identification, global exponential stability, state reconstructor, Sylvester equation.

Introduction

The basic problem dealt with in output regulation is to design a feedback controller which internally stabilizes a given linear time-invariant plant such that the output of the resulting closed-loop system converges to, or tracks, a certain reference signal of known frequencies in the presence of external disturbances of known frequencies. Since the performance of control system is mainly determined by its closed-loop poles, pole assignment is effective approach for designing feedback control systems. If all of the state variables of a time-invariant system are completely controllable and measurable, closed-loop poles of the system can be placed at any desired locations on the complex plane. However it is often very hard to perform conventional procedure becouse of several factors, like the presence of nonlinearity and unknown variations of operating conditions.

Artificial neural networks can be easily trained to performing a particular function. Becouse of that neural networks provide an effective on-line dynamic approach for synthesizing linear control systems via pole assignment. If the state variables measuremant is not possible or it is very slow or expensive neural network can be used for system identification.

Neural networks

All artificial neural networks are built of simple procesing elements called artificial neurons. Typical multiple-input neuron is shown in Figure 1. The individual inputs $p_1, p_2 \ldots p_R$ are each weighted by corresponding elements $w_1, w_2 \ldots w_R$ of the weight matrix $\boldsymbol{W}$. Exept inputs, the neuron has bias (offset) b which is sumarized with the inputs to form net input:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \cdots + w_{1,R}p_R + b = Wp + b \tag{1}$$

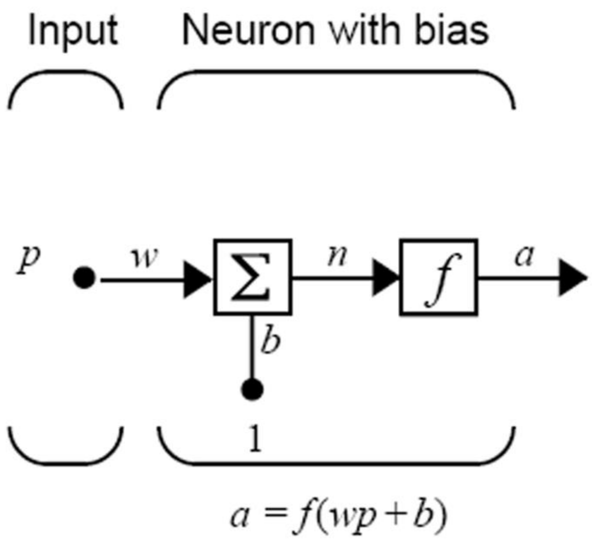

Figure 1. Typical multiple-input neuron.

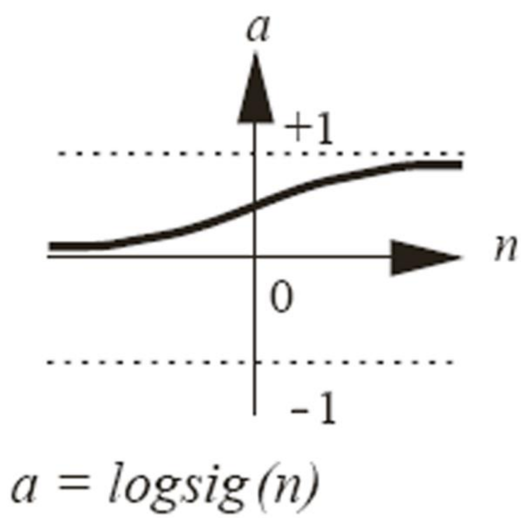

Figure 2. Log-sigmoid transfer function

We have adopted particular convenction in assigning the indicis of the elements of the weight matrix.The first index indicates the particular neuron destination and the second the source of the signal fed to it. Matrix $\boldsymbol{W}$ has only one row for single neuron and p is input vector. The net input goes into transfer function (activation function) f wich produces scalar output:

$$\mathrm{a} = \mathrm{f}(\mathrm{Wp} + \mathrm{b}) \tag{2}$$

A particular transfer function f is chosen to satisfy some specification of the problem that the neuron attempts to solve. One of the most commonly used functions is the log-sigmoid transfer function which is shown in Figure 2.

This transfer function takes the input (which may have any value between plus and minus infinity) and squashes the output into the range 0 to 1, according to the expression:

$$a = \frac{1}{1 + e^{-n}} \tag{3}$$

If we want to draw networks with several neurons, each have several inputs it might appear very complex if all lines are drawn. Thus, we wil use abbriviated notation showed in Figure 3.

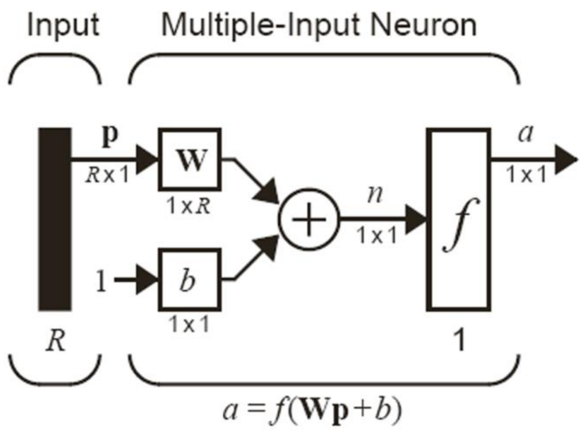

Figure 3. Multiple-input neuron in abbriviated notation.

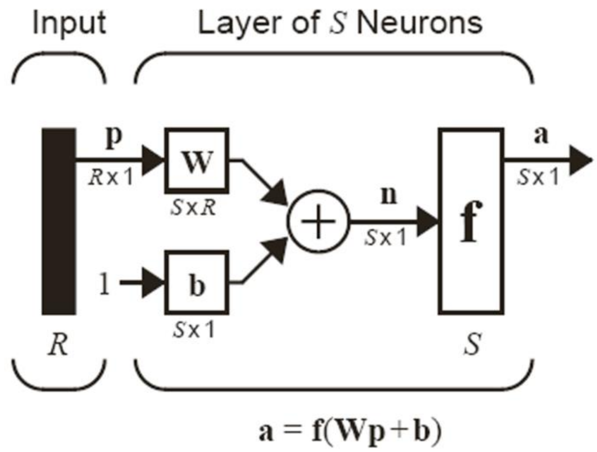

Figure 4. Single layer neural network of S neurons in abbriviated notation.

The input vector p is represented by the solid vertical bar. The dimension of p is displayed below the variable, indicating that input p is a single vector of R elements. These inputs go to the weight matrix, which has only one row in this single neuron case. A constant 1 multiplied by a scalar bias also enters the neuron as an input . The sum of the product Wp and bias b enters the transfer function f wich produces scalar output a.

Commonly one neuron, even with many inputs, may not be sufficient if we want to achieve complex behavior. We might need five or ten, operatingin parallel, in what we call layer. A single layer network of S neurons in abbriviated notation is shown in Figure 4. Each of the R inputs is connected to each of the neurons which means that weight matrix W has S rows. Each neuron has a bias, a summer, a transfer function boxes and an output . Taken together biases form the bias vector b and the outputs form the output vector a. It is possible that neurons in a single layer have different transfer functions.

Im most of the practical cases even a layer of neurons may not be sufficient to achieve wanted behavior. If it is a case when we apply multilayer neural network where several layers are connected in seres. Three-layer neural network in abbriviated notation is shown in Figure 5.

Each layer has its own weight matrix W, its own bias vector b, a net input vector n and output vector a. To distinguish between these layers we append the number of the layer as a superscript to the names for each of these variables. The outputs of layers one and two are the inputs for layers two and three. A layer which output is the network output is called an output layer. The other layers are called hidden layers.

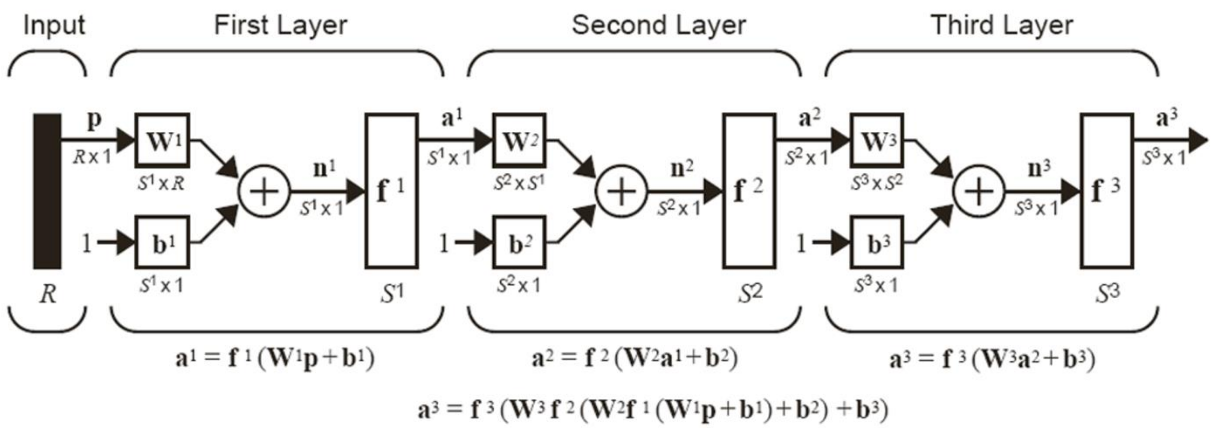

Figure 5. Three-layer neural network in abbriviated notation.

Two-layer networks, with sigmoid transfer functions in the hidden layer and linear transfer functions in the output layer, are universal approximators. We could use such networks to approximate almost any function, if we had a sufficient number of neurons in the hidden layer. The procedure for selecting the parameters (weights and biases) which will best approximate a given function for a given problem is called training the network.

Recurent neural network is a class of neural network with feedback. This feedback embodies short-time memory. A state layer is updated not only with the external input of the network but also with activation from the previous forward propagation. The feedback is modified by a set of weights as to enable automatic adaptation through learning (Hush, Abdallah and Horne 1981; Schmid; Leblebici; Mlynek. 1998).

Synthesizing Linear Feedback Control Systems Via Pole Assignment

When all of the state variables of a time-invariant system are completely controllable and measurable, the closed-loop poles of the system can be placed at any desired locations on the complex plane with state feedback through appropriate gains (Saberi 2003). Since the performance of control system is mainly determined by its closed-loop poles, pole assignment is effective approach to designing feedback control systems, especially for multivariate systems.

Consider a controllable linear time-invariant system as follows:

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad x(0) = x_0 \tag{4}$$

where $x \in R^n$ is state vector, $u \in R^m$ is control vector and $A \in R^{n\times n}$ and $B \in R^{n\times m}$ are known coefficient matrices. If linear state feedback control law applied to the system is

$$u(t) = r(t) + Kx(t) \tag{5}$$

then, colosed-loop sistem have following form

$$\dot{x}(t) = (A + BK)x(t) + Br(t), \qquad x(0) = x_0 \tag{6}$$

where $r \in R^m$ is reference input vector, and $K \in R^{m\times n}$ is state feedback gain matrix. According to the linear system theory, the matrix K may be evaluated by solving Sylvester equation (Benner 2004)

$$AZ - Z\Lambda = -BG \tag{7}$$

$$KZ = G \tag{8}$$

for fixed $\Lambda \in R^{n\times n}$ and almost any $G \in R^{m\times n}$. It has been proven that, if Λ is cyclic and has prescribed eigenvalues, A and Λ have no common eigenvalue ($\sigma(A) \cap \sigma(B) = \emptyset$), (G, Λ) is observable, then: the unique solution Z of Sylvester equation is almost surely nonsingular with respect to the parameter G, and the spectrum of $(A + BK)$ equals that of Λ.

The usual procedure for computing matrix K has two phases:

1. Choose matrix $\Lambda \in R^{n\times n}$ with prescribed eigenvalues, which should be cyclic and should not have common eigenvalues with $A(\sigma(A) \cap \sigma(B) = \emptyset)$, matrix G which should be observable with Λ, and solve Sylvester equation with respect to Z.
2. Solve second Sylvester equation in respect with K.

Neural network for control syntheses

Many numerical algorithms are prpoposed in control system theory for solving Sylvester equation. But such algorithms may not be efficient enough when applying to large-scale feedback control systems or online solving the time-varying feedback gain in adaptive control systems. In recent years, recurrent neural networks have been proposed for synthesizing linear control systems through pole assignment.

A couple of recurrent neural networks for pole assignment is proposed (Hanzalek 2005) with the following dynamic equations

$$\frac{dZ(t)}{dt} = -\mu_z\{A^T F_z[AZ(t) - Z(t)\Lambda + BG] - F_z[AZ(t) - Z(t)\Lambda + \mathrm{BG}]\Lambda^T\} \tag{9}$$

$$\frac{dK(t)}{dt} = -\mu_k\{F_k[K(t)Z(t) - G]Z(t)^T\} \tag{10}$$

where F_z and F_k are matrices of nondecreasing activation functions, μ_z and μ_k are positive scaling constants (design parameters), $Z(t)$ is an $n \times n$ activation state matrix coresponding to matrix Z in Sylvester equation, $K(t)$ is and $n \times m$ activation state matrix corresponding to the feedback gain matrix K in Sylvester equation. The same equations could be writen in abbriviated form

$$\frac{dZ(t)}{dt} = -\mu_z[A^T U(t) - U(t)\Lambda^{\mathrm{T}}] \tag{11}$$

$$U(t) = F_z[AZ(t) - Z(t)\Lambda + \mathrm{BG}] \tag{12}$$

$$\frac{dK(t)}{dt} = -\mu_k V(t)Z(t)^T \tag{13}$$

$$V(t) = F_k[K(t)Z(t) - G] \tag{14}$$

As we can see network is composed of two layes one hidden and one output layer. Hidden layer is used for evaluating of matrix Z. It is composed of two sublayers, hidden sublayer and output sublayer and each of them of $n \times n$ neurons. Basic memory matrix for the hidden sublayer (of the hidden layer) neurons is defined as product BG. Ther is no memory matrix for the output sublayer (of the hidden layer). Output layer is used for evaluating of matrix K. It is also composed of two sublayers, each of them of $n \times m$ neurons. There are no connections between neurons of the same sublayer. The biasing treshold matrix for the hidden sublayer is defined as $-G$ and this matrix is also not defined for the output sublayer (of the output layer). Signal flow of dynamic pole assignment process using neural network is ilustrated in Figure 6. If we look second Sylvester equation we can realized that output layer can be separated in n independet networks, each of them related to particular row of matrix K. In the same equation matrix G is the same for all rows. That means that output layer can be realized using one subnetwork with time-sharing memory. In that case the subnetwork compute different row of matrix K in different time sequences.

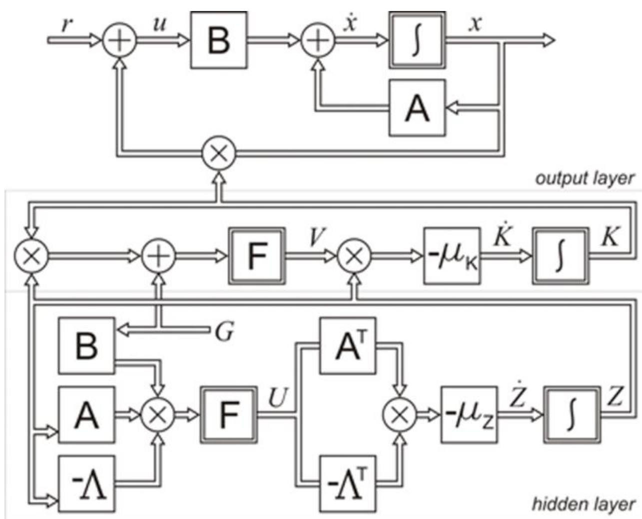

Figure 6. Signal flow of dynamic pole assignment process using neural network.

Conventional procedures for controler synthesis via pole assignment mean that matrices Z and K are computed offline and then control is aplied. But many real systems have unstable parameters which means that sistem can reach unstable zone. In that case we can ensure stability using above presented neural network which provide self tuning of control parameters as respond to system parameters changing.

This kind of control systems is called adaptive systems with feedback control law

$$u(t) = r(t) + K(t)X(t), \qquad K(0) = K_0 \tag{15}$$

where $K(t)$ is time variant state feedback gain matrix.

Global exponential stability

Above described two layer neural network for Sylvester equation solving is composed of generic neural network. This kind of network has proven ability for solving linear matrix equation. Stable value of state feedback gain matrix $K(t)$ is constant matrix K.

$$K = \lim_{k \to \infty} K(t) \tag{16}$$

Dynamic pole assignment using neural networks has complex time scale. Short-term memory $K(t)$ of the output layer represent long-term memory (state feedback gain matrix) of the controller. Short-term memory of the control system is state vectore $x(t)$.

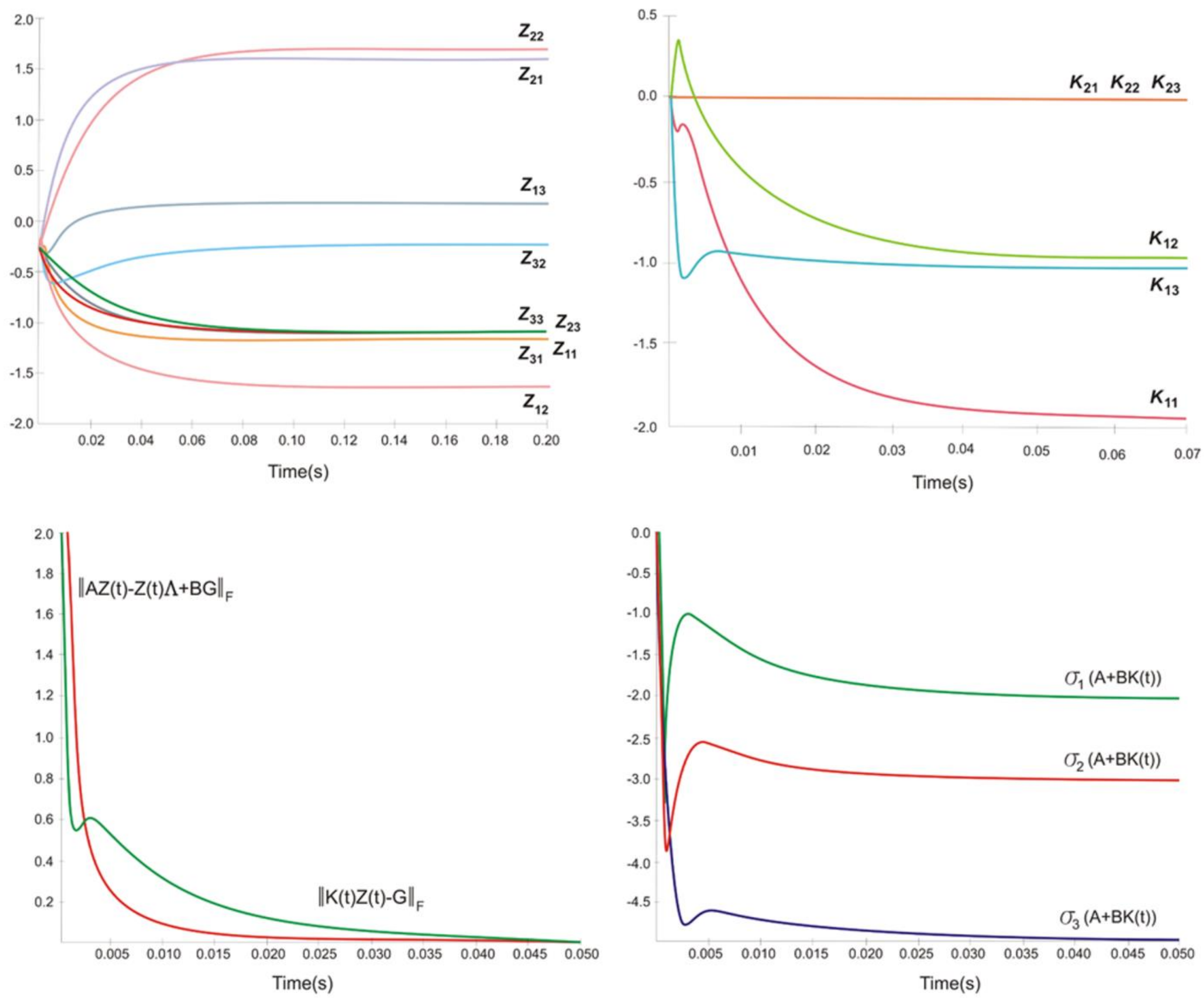

Figure 7. Simulation results of pole assignment process using neural network.

Short-term memory $Z(t)$ of the hidden layer is, at the same time, long-term memory (weight matrix) of the output layer. Convergence rate of input and output layer are proportional to design parameters μ_z and μ_k. In Figure 7, simulation results are given to verify theoretical results discussed in previous previous sections. We can se the transient states $Z(t)$ and $K(t)$ of recurrent neural networks, and dynamical process of $\|AZ(t) - Z(t)\Lambda + \mathrm{BG}\|_F$, $\|K(t)Z(t) - \mathrm{G}\|_F$ and $\sigma_i\big(A + BK(t)\big), i = 1,2,3,$ where i denotes the resulting closed-loop poles corresponding to the th eigenvalue of Λ. The simulated recurrent neural system yields the exact pole assignment solution.

Conclusions

Classical aproach for synthesizing control systems via pole assignment is appropriate only for time invariant systems. But, if the paramters are not constant system can easily reach unstable zone. The recurrent neural networks, proposed in previous sections, provide an effective on-line dynamic approach to synthesizing linear control systems via pole assignment. To apply this method, it is necessary that system must be complitly controllable and observable.

State variables of real control systems can be inconveniend for state feedback control. Main reason is that measuremant is not possible or it is very slow or expensive. It is the case when state reconstructor is used. Reconstructor is device which determine unknown state variables based on input vector and known output variables. For linear systems this process is reduced to output feedback gain matrix determination. This matrix is mostly time-invariant but not for systems with unstable parameters. In the case of time-invariant systems it is necessary that output feedback gain matrix is on-line computed (Hsin Yu 1996). One of the most effective vay to achieve this task is using two-layer neural network (Kim, Fok, Fregene, Hoon Lee, Seok Oh and Wang 2004).

More complex, four-layer, neural networ are designed to achieve optimal control based on given criterion (McDermott and Athans 1994). The criterion can be minimal average level of state matrix or minimal level of state feedback gain matrix. In the case of state reconstructor four-layer neural networks are used for on-line computing of output feedback gain matrix with minimal average level of it’s elements.

References

[1] Ali Saberi. 2003: Control of linear systems with regulation and input constraints. School of Electrical Engineering and Computer Science, Washington State University

[2] Peter Benner. 2004: Factorized Solution of Sylvester Equations with Applications in Control. *Sign (H)* Vol. 1.

[3] Don Hush; Chaouki Abdallah; Bill Horne, 1991: The Recursive Neural Network. Department of Electrical Engineering and Computer Engineering. University of New Mexico, Albuquerque, U.S.A.

[4] Z.Hanzalek, 2005: Parallel Algorithm for Feedforward Neural Network Controller. Trnka Laboratory for Automatic Control, Department of Control Engineering, Czech Technical University in Prague, Czech Republic.

[5] McDermott, W.E.; Athans, M. 1994: Approximating Optimal State Feedback Using Neural Networks. Department of Electrical Engineering and Computer Science Massachusetts Institute of Technology. *Proceedings of th 33rd Institute of Electrical and Electronic publications* Vol. 3.

[6] Ssu-Hsin Yu. 1996: Model-based identification and control of nonlinear dynamic systems using neural networks. Department of mechanical engineering Massachusetts Institute of Technology.

[7] Il-Hwan Kim; Stanley Fok; Kingsley Fregene; Dong-Hoon Lee; Tae-Seok Oh; David W. L. Wang. 2004: Neural Network-Based System Identification and Controller Synthesis for an Industrial Sewing Machine. *International Journal of Control, Automation, and Systems* Vol. 2.

[8] Alexandre Schmid; Yusuf Leblebici; Daniel Mlynek. 1998: Hardware realization of a hamming neural network with on-chip learning, Swiss Federal Institute of Technology, Integrated Systems Center, Lausanne, Switzerland.