Quantization Effect Influences Identification in Adaptive LQ Controller

Kamil Švancara, Petr Pivoňka
Department of Control and Instrumentation
Brno University of Technology
Božetěchova 2, 612 66 Brno, Czech Republic
Tel.: +420/541 141 172 Fax: +420/541 141 123
Email: svancara@feec.vutbr.cz, pivonka@feec.vutbr.cz

***Abstract*— Many controlling algorithms were developed to satisfy designer's assumption that controlled process is stable, optimal, adaptive etc. Only few of them should be used in the real-time, on-line adaptation or implemented into Programmable Logic Controller. Algorithms based on neural networks were successfully tested as for the process control as for the process identification. In simulation, quantization effect given by A/D and D/A converter is very often left out. Quantization effect influences identified process transfer function and controller possibilities. In this case, controller controls the process inaccurate and often with oscillations than without quantization. This paper shows a comparison between two identification methods. On-line identification (in the real time) based on neural networks and a classical identification are implemented in adaptive LQ controller with three types of A/D and D/A converters at least be closer in simulation the real process controlling.**

***Index Terms*— Adaptive controller, optimal controller, ARMA model, LD-FIL decomposition, Matlab.**

I. Introduction

Two methods for identification are used. Both of them satisfied assumptions: future implementation into PLC and real-time solution. The first one, algorithm based on neural networks is used in comparison with the classical RLS method augmented by LD-FIL matrix decomposition. LD-FIL as a classical robust algorithm could be used mainly for its attributes: numerically stable algorithm and easy implementation for on-line solution. Neural networks were used with highly popular error back-propagation learning algorithm. This paper is not focused on identification as a stand-alone part without controlling, but inseparable part of the adaptive LQ controller.

ARMA model is identified in the closed loop. Identified transfer function is used in each step in LQ controller. LQ controller's action value, input to the process, is solved in the real-time. In regulation, the step response is not the only single scale for the measurement of the control quality. Disturbance cancellation and curse of the action value are other powerful scales. Both identification methods for three different quantization effects are compared further in the article. In conclusion, advantages and disadvantages of the identification based on neural networks in comparison with the classical identification are presented [4].

II. Process Identification

ARMA model without coefficient b_0 for identification is used. ARX should be used too, but this model could enough represent identified process as can be seen in (1)

$$F_{\mathrm{M}}(z^{-1}) = \frac{b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3}}{1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3}} = \frac{Y_{\mathrm{M}}(z^{-1})}{U_{\mathrm{M}}(z^{-1})} \quad (1)$$

A. Classical Identification with LD-FIL Matrix Decomposition

Recursive Least mean Square (RLS) method is used for on-line identification in many articles and papers [2]. Usual form with the vector of parameters θ_{i+1} and covariance matrix $\mathbf{C}_{i+1}$ could be written as

$$\theta_{i+1} = \theta_i + \mathbf{C}_{i+1}\varphi_{i+1}\left(y_{i+1} - \varphi_{i+1}^{\mathrm{T}}\theta_i\right) \quad (2)$$

$$\mathbf{C}_{i+1} = \mathbf{C}_i - \mathbf{C}_i\varphi_{i+1}\left(1 + \varphi_{i+1}^{\mathrm{T}}\mathbf{C}_i\varphi_{i+1}\right)^{-1}\varphi_{i+1}^{\mathrm{T}}\mathbf{C}_i \quad (3)$$

where i denotes the discrete time, φ_{i+1} denotes vector of inputs and outputs and y_{i+1} denotes current output.

ARMA model should be rewritten to mathematical model $y_{\mathrm{M}} = \varphi_{i+1}^{\mathrm{T}}\theta_i + \epsilon$ where φ_{i+1} is often called the regression vector and ϵ is the error. The parameter vector is solved to minimize least-square loss function in step i

$$\theta_i = (\Phi_i^{\mathrm{T}}\Phi_i)^{-1}\Phi_i^{\mathrm{T}}\mathbf{Y}_i = \mathbf{C}_i\Phi_i^{\mathrm{T}}\mathbf{Y}_i \quad (4)$$

where Φ_i^{T} denotes regression matrix and $\mathbf{Y}_i$ denotes output vector. Next step $i+1$

$$\theta_{i+1} = \mathbf{C}_{i+1}[\Phi_i^{\mathrm{T}}, \varphi_{i+1}][\mathbf{Y}_i, y_{i+1}]^{\mathrm{T}} \quad (5)$$

where $\mathbf{C}_{i+1}$ is given by

$$\begin{aligned}\mathbf{C}_{i+1} &= [\Phi_i^{\mathrm{T}}\Phi_i + \varphi_{i+1}\varphi_{i+1}^{\mathrm{T}}]^{-1} \\ &= \mathbf{G}_{i+1}\mathbf{D}_{i+1}\mathbf{G}_{i+1}^{\mathrm{T}}\end{aligned} \quad (6)$$

LD-FIL (lower-diagonal-upper) decomposition algorithm could be used in form as it is illustrated in (7)

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ g_{12} & 1 & 0 \\ g_{13} & g_{23} & 1 \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \begin{bmatrix} 1 & g_{12} & g_{13} \\ 0 & 1 & g_{23} \\ 0 & 0 & 1 \end{bmatrix} \quad (7)$$

where $\mathbf{G}_{i+1}$ denotes lower triangular matrix, $\mathbf{G}_{i+1}^{\mathrm{T}}$ denotes upper triangular matrix and $\mathbf{D}_{i+1}$ denotes diagonal matrix.

Parameters on the main diagonal mainly influence identification.

Well-known LD-FIL matrix decomposition is derived by lemma for matrix inversion (see 8) $\mathbf{G}_{i+1}$ denotes lower-triangular matrix

$$(\mathbf{A}+\mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B})^{-1}\mathbf{DA}^{-1} \quad (8)$$

then

$$\begin{aligned}\mathbf{G}_{i+1}\mathbf{D}_{i+1}\mathbf{G}_{i+1}^{\mathrm{T}} &= \mathbf{G}_i\mathbf{D}_i\mathbf{G}_i^{\mathrm{T}} - \mathbf{G}_i\mathbf{D}_i\mathbf{G}_i^{\mathrm{T}}\varphi_{i+1}(1+\varphi_{i+1}^{\mathrm{T}}\cdots \\ &\quad \mathbf{G}_i\mathbf{D}_i\mathbf{G}_i^{\mathrm{T}}\varphi_{i+1})^{-1}\varphi_{i+1}^{\mathrm{T}}\mathbf{G}_i\mathbf{D}_i\mathbf{G}_i^{\mathrm{T}} \\ &= \mathbf{G}_i\big[\mathbf{D}_i - \mathbf{D}_i\mathbf{G}_i^{\mathrm{T}}\varphi_{i+1}(1+\varphi_{i+1}^{\mathrm{T}}\mathbf{G}_i\mathbf{D}_i\cdots \\ &\quad \mathbf{G}_i^{\mathrm{T}}\varphi_{i+1})^{-1}\varphi_{i+1}^{\mathrm{T}}\mathbf{G}_i\mathbf{D}_i\big]\mathbf{G}_i^{\mathrm{T}} \\ &= \mathbf{G}_i\left[\mathbf{D}_i - \mathbf{D}_i\mathbf{f}_i\mathbf{f}_i^{\mathrm{T}}\mathbf{D}_i\frac{1}{1+\mathbf{f}_i^{\mathrm{T}}\mathbf{D}_i\mathbf{f}_i}\right]\mathbf{G}_i^{\mathrm{T}} \quad (9)\end{aligned}$$

where an auxilary vector $\mathbf{f}_i$ is given $\mathbf{f}_i = \mathbf{G}_i^{\mathrm{T}}\varphi_{i+1}$ [3], [5]. The third order ARMA model is used.

B. Identification Based on Neural Networks

The third order ARMA model is used in identification based on neural networks. Neural network should be connected with process through its input and output only. Neural network dynamics is represented by step delays as Fig. 1 shows [4].

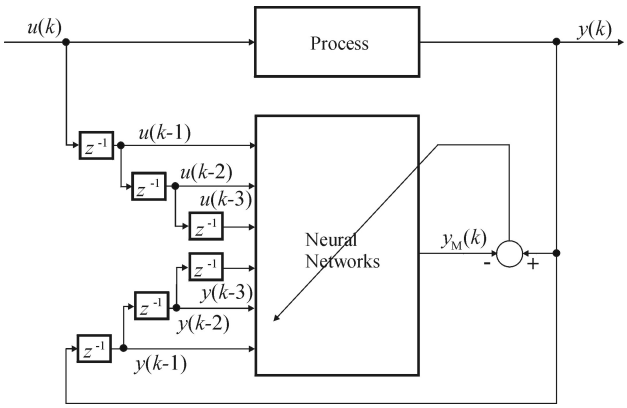

Fig. 1. The layout of identification based on Neural Networks.

Now, the back-propagation learning algorithm is used to identify transfer function [1]. On-line algorithm works for pattern-by-pattern updating of weights. Algorithm consists of cycles where initialization of weights should start. Random values with physical meaning instead of totally random values should be used for the weights. For example, the weight representing a_1 should be negative or the sum of b_1, b_2 and b_3 should be positive. Next step is given by on-line updating of training examples. The algorithm for choice of samples should be used to improve identified model [7].

Forward computation is the next step where the internal activity of each neuron j in layer l is given by

$$v_j^{(l)}(n) = \sum_{i=0}^{p} w_{ji}^{(l)}(n)\, y_i^{(l-1)}(n) \quad (10)$$

where $y_i^{(l-1)}(n)$ is the function signal of neuron i in the previous layer $(l-1)$ at iteration n and $w_{ji}^{(l)}(n)$ is the synaptic weight of neuron j in layer l. Number p denotes inputs. The output function of neuron j in layer l is sigmoid

$$y_j^{(l)}(n) = \frac{2}{1+\exp(-v_j^{(l)}(n))} - 1 \quad (11)$$

or identity

$$y_j^{(l)}(n) = v_j^{(l)}(n) \quad (12)$$

Hence, the error signal in the output layer (i.e., $l = L$) is

$$e_j(n) = d_j(n) - y_i^{(L)}(n) \quad (13)$$

where $d_j(n)$ is the j-th element of the desired signal. The weights are solved to minimize sum of squared errors

$$\begin{aligned}\frac{\partial\mathcal{E}(n)}{\partial w_{ji}(n)} &= \frac{\partial\mathcal{E}(n)}{\partial e_j(n)}\frac{\partial e_j(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)}\frac{\partial v_j(n)}{\partial w_{ji}(n)} \\ &= -e_j(n)\frac{\partial y_j(n)}{\partial v_j(n)}y_i(n) \quad (14)\end{aligned}$$

Next step is given by backward computation where local gradients $\delta = -\partial\mathcal{E}(n)/\partial w_{ji}(n)$ is given by

$$\delta_j^{(L)}(n) = e_j^{(L)}(n)y_j^{(L)}(n)\left[1 - y_j^{(L)}(n)\right] \quad (15)$$

for output layer L and sigmoidal output function and

$$\delta_j^{(l)}(n) = y_j^{(l)}(n)\left[1 - y_j^{(l)}(n)\right]\sum_k \delta_k^{(l+1)}(n)w_{kj}^{(l+1)}(n) \quad (16)$$

for hidden layer l. The synaptic weights of the network in layer l are adjusted according to generalized delta rule (17)

$$\begin{aligned}w_{ji}^{(l)}(n+1) &= w_{ji}^{(l)}(n) + \alpha\left[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)\right] \\ &\quad + \eta\delta_j^{(l)}(n)y_i^{(l-1)}(n) \quad (17)\end{aligned}$$

where α is the momentum constant and η is the learning-rate parameter. These parameters represent algorithm's adjusting parameters with influence on the rate of convergence. Parameters should be limited to interval $(0, 1)$.

III. Adaptive LQ Controller

LQ controller is solved according to identified ARMA model in each step and according to minimization of the quadratic performance [3], [5] and [6]. Identification ensures adaptation in the real time. Quadratic performance is defined by

$$J = \mathbf{x}_T^{\mathrm{T}}\mathbf{Q}\mathbf{x}_T + \sum_{i=i_0+1}^{i_0+T} q_y(w_i - y_i)^2 + q_u(u_i - u_i^0)^2 \quad (18)$$

where w_i denotes desired value, y_i denotes output of the process, u_i denotes action value, u_i^0 denotes action value for offset elimination and it is equal to desired value. Parameter q_y (q_u) denotes weight for process output (input), i_0 denotes the first step while the minimization is used and denotes the minimum at the last step $i_0 + T$. Fig. 2 shows model in MATLAB/Simulink.

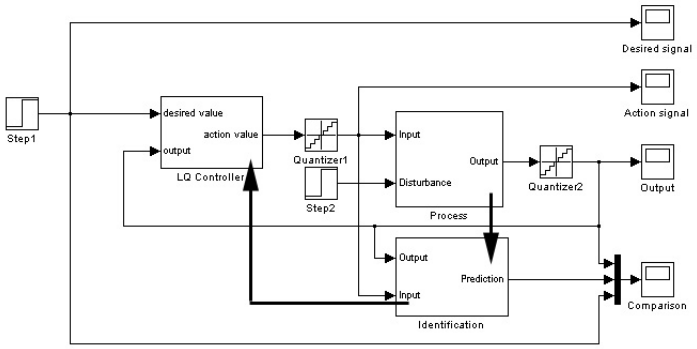

Fig. 2. Simulation model in MATLAB/Simulink.

The quadratic performance can be rewritten to more suitable form

$$J = \sum_{i=i_0+1}^{i_0+T} \mathbf{z}_i^{\mathrm{T}} \mathbf{Q} \mathbf{z}_i, \tag{19}$$

where $\mathbf{z}_i^{\mathrm{T}} = [\mathbf{x}_i, w_i, u_i^0]$ and $\mathbf{x}_i^{\mathrm{T}} = \mathbf{S}_i[u_i, \mathbf{x}_{i-1}, w_i, u_i^0] = \mathbf{S}_i \mathbf{z}_{i-1}$ and weight matrix $\mathbf{Q}$ is more universal. The weight matrix (22) is implemented to the quadratic performance in equations (23), (24) and (26). This means the matrix can realize the quadratic performance and even more as an incremental weighting or an integral action. We will work with pseudo-state matrix $\mathbf{S} = [\mathbf{S}_u, \mathbf{S}_x, \mathbf{S}_w, \mathbf{S}_{u^0}]$ defined by equations (20) and (21) where for our example, model's order is $n = r = 3$ and the coefficient $b_0 = 0$.

$$\begin{aligned} \mathbf{S}_u &= \begin{bmatrix} 1 & 0 & 0 & b_0 & 0 & \dots & 0 \end{bmatrix}^{\mathrm{T}} \\ \mathbf{S}_w &= \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}^{\mathrm{T}} \\ \mathbf{S}_{u^0} &= \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{20}$$

$$\mathbf{S}_x = \begin{bmatrix} 0 & 0 & 0 \dots & 0 & 0 \\ 1 & 0 & 0 \dots & 0 & 0 \\ 0 \dots & 0 \dots & 0 & 0 \\ b_1 \dots & b_r & a_1 \dots & a_n \\ 0 \dots & 0 & 1 \dots & 0 \\ \vdots & \ddots & \vdots & 0 & \ddots & 0 \\ 0 \dots & 0 & 0 \dots & 0 \end{bmatrix} \tag{21}$$

Universal weight matrix should be written in many forms to designer's expectation [5], [6]. One example of universal matrix shows equation (22)

$$\mathbf{Q} = \begin{bmatrix} q_u & -q_u & 0 \dots & 0 & \dots & 0 & -q_u \\ -q_u & q_u & 0 \dots & 0 & \dots & 0 & 0 \\ 0 & 0 & q_u \dots & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 \dots & q_y & 0 & -q_y & 0 \\ \vdots & \vdots & \vdots \ddots & 0 & q_y & -q_y & 0 \\ 0 & 0 & 0 \dots & -q_y & -q_y & q_y & 0 \\ -q_u & 0 & 0 \dots & 0 & \dots & 0 & q_u \end{bmatrix} \tag{22}$$

The method for the minimization in the each step ahead is known and it is described at the next equations (23), (24) and (26)

$$J_{i_0} = \mathbf{z}_{i_0+T-1}^{\mathrm{T}} \mathbf{S}^{\mathrm{T}} \mathbf{Q} \mathbf{S} \mathbf{z}_{i_0+T-1} = \mathbf{z}_{i_0+T-1}^{\mathrm{T}} \mathbf{H} \mathbf{z}_{i_0+T-1} \tag{23}$$

TABLE I

LQ CONTROLLER SOLVED IN ITERATION ALGORITHM

Step	Equation	Notes
1.	$\mathbf{H}^* = \mathbf{H}_{xx} - \mathbf{H}_{ux}^{\mathrm{T}} H_{uu}^{-1} \mathbf{H}_{ux}$	recursively solves lost function
2.	$\mathbf{G}\mathbf{D}\mathbf{G}^{\mathrm{T}} = \mathbf{H}$	LD-FIL decomposition
3.	$u_i = -G_{uu}^{-1} \mathbf{G}_{ux} \mathbf{x}_{i-1}$	solves action value

and the minimum is given for derivation by u_{i_0+T} equal to zero at last step $i_0 + T$

$$\begin{aligned} min[\Psi_{i_0+T}] = \bar{\mathbf{z}}_{i_0+T-1}^{\mathrm{T}} \big(& \mathbf{H}_{xx} \\ & - \mathbf{H}_{ux}^{\mathrm{T}} H_{uu}^{-1} \mathbf{H}_{ux} \big) \bar{\mathbf{z}}_{i_0+T-1} \end{aligned} \tag{24}$$

where $\bar{\mathbf{z}}_{i_0+T-1}^{\mathrm{T}}$ is $\mathbf{z}_{i_0+T-1}^{\mathrm{T}}$ without u_{i_0+T-1} and

$$\mathbf{H} = \begin{bmatrix} H_{uu} \mathbf{H}_{ux} \\ \mathbf{H}_{xu} \mathbf{H}_{xx} \end{bmatrix} \tag{25}$$

We can simply see that $\mathbf{H}$ is the symmetric matrix and consecutively the next minimization step $i_0 + T - 1$ is defined by

$$\begin{aligned} min[\Psi_{i_0+T}] = \bar{\mathbf{z}}_{i_0+T-1}^{\mathrm{T}} \mathbf{H}^* \bar{\mathbf{z}}_{i_0+T-1} \\ + \mathbf{z}_{i_0+T-1}^{\mathrm{T}} \mathbf{Q} \mathbf{z}_{i_0+T-1} \end{aligned} \tag{26}$$

where matrix $\mathbf{H}^* = \mathbf{H}_{xx} - \mathbf{H}_{ux}^{\mathrm{T}} H_{uu}^{-1} \mathbf{H}_{ux}$ is defined at step $i_0 + T$. Using LD-FIL decomposition (see (7)) for matrix $\mathbf{G}$ instead of $\mathbf{H}$, where $\mathbf{H} = \mathbf{G}\mathbf{D}\mathbf{G}^{\mathrm{T}}$ we can rewrite the quadratic performance to the triangular factor quadratic norm

$$\|\mathbf{G}[u_i, \mathbf{x}_{i-1}, w_i, u_i^0]^{\mathrm{T}}\|^2 \tag{27}$$

Now, it is simple to find control low u_i with influences on the first row only of the minimization at step i

$$G_{uu} u_i + \mathbf{G}_{ux} \mathbf{x}_{i-1} = 0 \tag{28}$$

where the minimum is given $\|\mathbf{G}_{xx} \mathbf{x}_{i-1}\|^2$ and G_{uu}, $\mathbf{G}_{ux}$ and $\mathbf{G}_{xx}$ are sub-matrices of $\mathbf{G}$. Finally, the control low is given by

$$u_i = -G_{uu}^{-1} \mathbf{G}_{ux} \mathbf{x}_{i-1} \tag{29}$$

LQ is solved at the each one step ahead. Table I shows recursively solved LQ iteration algorithm.

IV. SIMULATION RESULTS

Comparison of both type identifications and the potential consequences of the skipped quantization effect are shown in the simulation experiment on the process with transfer function

$$F_{\mathrm{S}}(s) = \frac{2}{(10s+1)(s+1)^2} \tag{30}$$

where disturbance enters between process main dynamics F_{S1} and additional, faster dynamics F_{S2} (e.g. servomechanism)

$$F_{\mathrm{S1}}(s) = \frac{2}{10s+1} \tag{31}$$

$$F_{\mathrm{S2}}(s) = \frac{1}{(s+1)^2} \tag{32}$$

Fig. 3–10 show the process response and disturbance cancellation together with controller action value. Figures show

both identification methods. The process output (every upper sub-figures) and input (every lower sub-figures) are shown for desired step set to $+2$ V at time 20 s. Deterministic disturbance step set to $+1$ V at time 80 s. The sampling period was set to $T_{\mathrm{S}} = 1$ s at the beginning. In Fig. 3, the classical RLS identification in closed loop with LQ controller is shown.

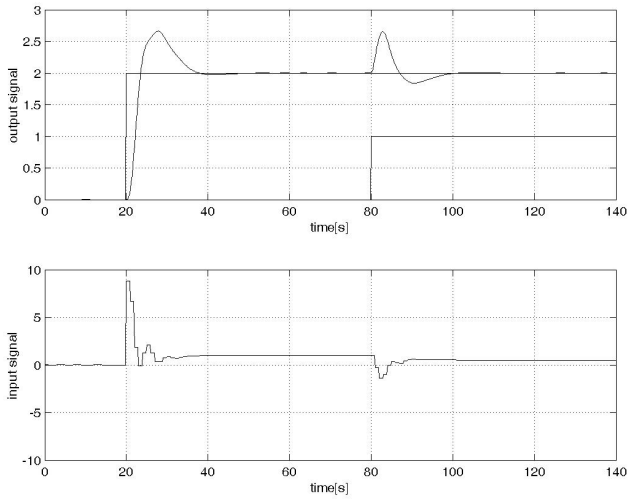

Fig. 3. Adaptive LQ controller with closed loop identification. The sampling period is $T_{\mathrm{S}} = 1$ s. LD-FIL decomposition is used. A/D and D/A converters are 12bits.

Simulation result shows that adaptive LQ controller could work well for the sampling period $T_{\mathrm{S}} = 1$ s, but provided the sampling period decreases it reduces overshot and solution time. Decreasing of the sampling period is very important for disturbance cancellation too. Because of this reason, the sampling period was set to $T_{\mathrm{S}} = 0.1$ s. Fig. 4 shows the RLS identification in closed loop with LQ controller. A/D and D/A converters are not used.

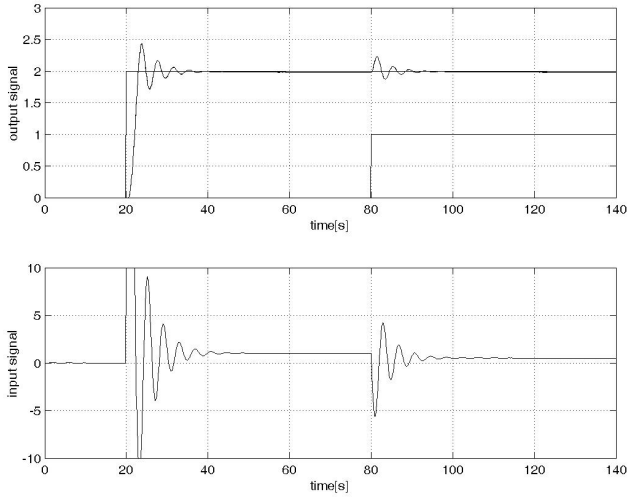

Fig. 4. Adaptive LQ controller with closed loop identification. The sampling period is $T_{\mathrm{S}} = 0.1$ s. LD-FIL decomposition is used. A/D and D/A converters are not used.

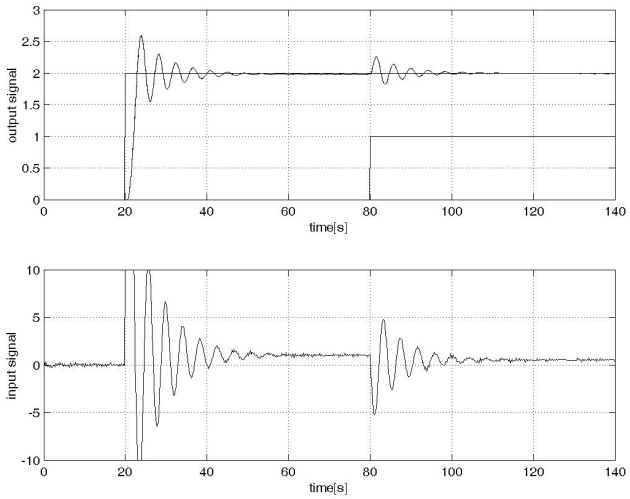

Fig. 5. RLS identification with LD-FIL decomposition in closed loop with LQ controller for 12bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

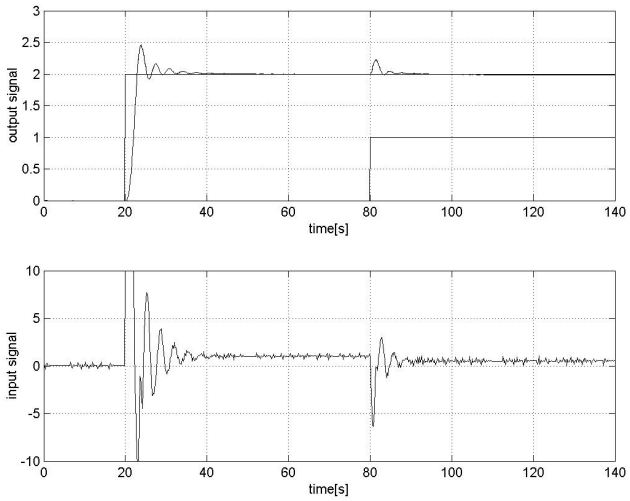

Fig. 6. Identification based on neural networks with the momentum constant $\alpha = 0.01$ and the learning-rate parameter $\eta = 0.001$ in closed loop with LQ controller for 12bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

The idea of decreasing sampling period has been already justified. Problem becomes when the simulation is extended with a quantization effect. A/D and D/A converters produce the quantization effect in each real control processes. The precise control input and process output values are reduced to imprecise values according type of A/D and D/A converters. For example, 12bits A/D or D/A converters limited to ± 10 V reduce values with 4 valid positions divisible by 0.0048 V.

Fig. 5, 7 and 9 show the RLS identification with LD-FIL decomposition in closed loop with LQ controller.

Fig. 6, 8 and 10 show the identification based on neural networks in closed loop with LQ controller.

Three quantization effects given by A/D and D/A converters are compared:

- 12bits A/D and D/A converters;

- 10bits A/D and D/A converters;
- 8bits A/D and D/A converters.

Input to the process is limited to ± 10 V. Both identifications are used for the same set-up of LQ controller with purpose of the equivalent comparison.

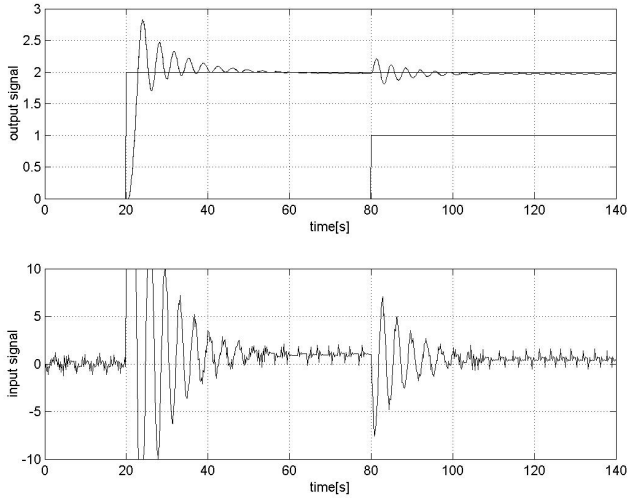

Fig. 7. RLS identification with LD-FIL decomposition in closed loop with LQ controller for 10bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

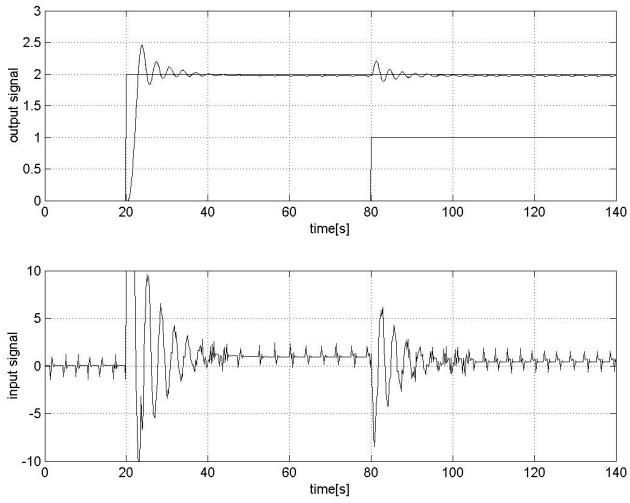

Fig. 8. Identification based on neural networks with the momentum constant $\alpha = 0.01$ and the learning-rate parameter $\eta = 0.001$ in closed loop with LQ controller for 10bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

V. Conclusions

The presented paper showed where the closed loop on-line identification based on neural networks could be better used than the classical identification. Both identification methods were used in adaptive LQ controller. In Fig. 3–10, the process output and input together with disturbance response are shown. The sampling period was decreased to $T_{\mathrm{S}} = 0.1$ s because

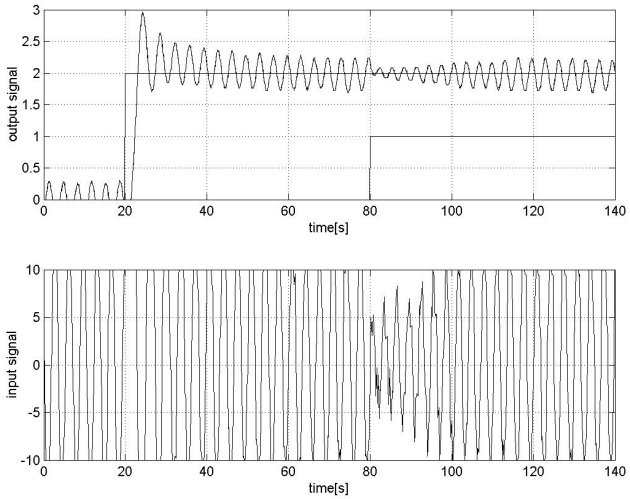

Fig. 9. RLS identification with LD-FIL decomposition in closed loop with LQ controller for 8bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

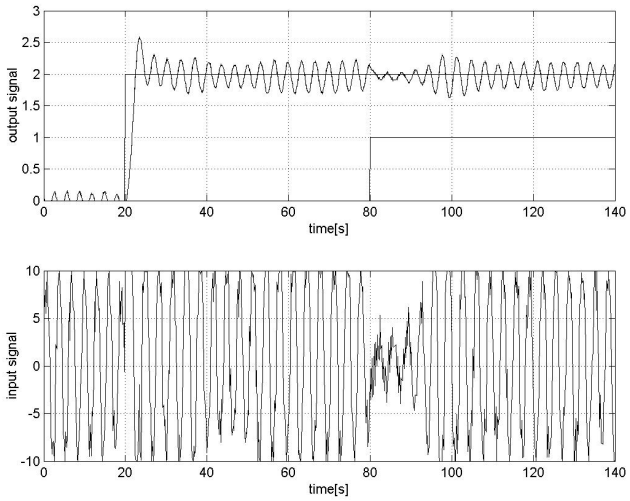

Fig. 10. Identification based on neural networks with the momentum constant $\alpha = 0.01$ and the learning-rate parameter $\eta = 0.001$ in closed loop with LQ controller for 8bits A/D and D/A converters. $T_{\mathrm{S}} = 0.1$ s.

of better disturbance cancellation. Figures show the control without quantization effect and three types of quantization effects, given by A/D and D/A converters, influences adaptive regulation. For 12bits A/D and D/A converters, both results are usable. Identification based on neural networks, according to its input and output, has been much better adapted than RLS identification with LD-FIL in comparison with 10bits converters. For 10bits, results are usable, but RLS identification with LD-FIL decomposition could not identify given system in adaptive LQ controller without oscillations of input. For 8bits A/D and D/A converters, both results are useless. Outputs oscillate.

Identification based on neural network with LQ controller works better according to input smoothness, output overshot etc. The reasons way the identification based on neural net-

works has been more successful could be explained in several ways:

1) error back-propagation algorithm works with the parameter momentum (the parameter momentum works as a filter to avoid oscillations but similar filter added to the classical controllers produces slower step response);
2) the classical identification sampling period cannot be too short otherwise identification transfer function loses the correct estimation of the real process;
3) the neural networks identification does not have to use every samples as classical identification [7].

On the other hand, identification based on neural networks needs initialization which is always problematic. Simulation results are solved in MATLAB/Simulink. Algorithms are prepared for implementation into PLC B&R. Identification RLS with LD-FIL matrix decomposition has been already successfully tested on the real physical models.

ACKNOWLEDGMENTS

This research was supported by the Czech Grant Agency GACR, under the grant number 102/01/1485 Environment for Developing, Modelling and Application of Heterogeneous Systems and by the MSMT research plan CEZ MSM: 260000013 Automation of Technological and Manufacturing Processes.

REFERENCES

[1] Simon Haykin, *Neural Networks, A Comprehensive Foundation*. Macmillan College Publishing Company, 1994.
[2] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems, Theory and Design*. Prentice Hall, 3.ed., 1997.
[3] Josef Böhm, "Survey in Using Square-roots factors," *AV CR, UTIA*, num. 1054, in Czech, 1980.
[4] Kumpati S. Narendra, "Neural Networks for Control: Theory and Practise," *Proceedings of The IEEE*, vol. 84, no. 10, pp. 1385-1406, 1996.
[5] Kamil Švancara, "Adaptive Optimal Controller Using Square-root Factors Implemented in PLC," *In Proceedings of 8th Conference Student EEICT*, pp. 99-103, 2002.
[6] K. Švancara and P. Pivoňka, "Closed Loop On-line Identification Based on Neural Networks in Adaptive Optimal Controller," *Proceedings of 10th Zittau Fuzzy Colloquium*, pp. 218-223, 2002.
[7] H. Vychodil, P. Pivoňka and P. Krupanský, "The Algorithm for Choice of Samples in Training Set for Neural Networks," *Proceedings of 10th Zittau Fuzzy Colloquium*, pp. 224-229, 2002.