Strojniški vestnik - Journal of Mechanical Engineering 54(2008)2, 94-102
UDK - UDC 007.52
Paper received: 20.3.2007
Paper accepted: 19.12.2007

Teleoperation of SCARA with Neural Network Based Controller

Jure Čas* - Rok Klobučar - Darko Hercog - Riko Šafarič
University of Maribor, Faculty of Electrical Engineering and Computer Science, Slovenia

This paper describes the development of neural network based controller for the teleoperation of Selective Compliance Assembly Robot Arm (SCARA). The SCARA is controlled and teleoperated via the internet. Presented experiment is used by students at the University of Maribor as a remote educational tool. Application is based on MATLAB/Simulink and LabVIEW software packages. MATLAB/Simulink and developed library DSP-2 Library for Simulink are used for neural network control algorithm development, simulation and code generation. The executable code is downloaded to the Digital Signal Processor (DSP). The DSP controls through the analog and digital I/O the real process and maintain the data connection with the laboratory server. The LabVIEW virtual instrument (VI) is used as a client server application for the teleoperation. LabVIEW VI provides the ability for parameter tuning, signal monitoring, on-line analysis and via Remote Panels technology also teleoperation by using the internet browser (Internet Explorer). The main advantage of a neural network controller is the exploitation of its self-learning capability. For example: when friction or an unexpected disturbance occurs, the user of a remote application does not need any information about the changed robot dynamics, because it is estimated independently of the remote user.

Keywords: robotics, neural network controller, remote experiment system, LabVIEW, MATLAB, Simulink

0 INTRODUCTION

The theoretical development of a continuous neural-network sliding-mode controller (CNNSMC) based on the theory of continuous sliding-mode controllers is presented in the paper. Derived equations of the CNNSMC were verified on the real laboratory teleoperated SCARA mechanism, which is used as remote educational tool. An inverse dynamic model of a robot arm mechanism is needed to design a robot controller but such a modelling of dynamic system is not an easy task. Besides that, the users of the remote robot experiment are unable to physically remove the disturbing causes of a robot's dynamics and, what is even more obstructive, the disturbance dynamics is completely unknown to "the user of the experiment". The consequence is that the conventional procedures for estimating of robot dynamics, which are used in non-teleoperation robot applications, can not be used for the purposes of teleoperating applications.

When using the CNNSMC as a control algorithm however, unknown robot dynamics is not a control problem anymore, because the approximation of changed robot dynamics is computed independently of the remote user. The additional advantage of application is the possibility for the on-line tuning parameters of CNNSMC, which improves the performance of a robot's controller and makes application more appropriate as an educational tool.

Students are able to learn how the parameters of CNNSMC have an affect on its performance because of the possibility of on-line tuning parameters and observing the responses of regulation using graphs and numerical indicators. They can improve their knowledge of the neural network control algorithm and, what is more important, with the use of internet connections experiments can be executed on a real SCARA from somewhere outside a laboratory .

The developed software, which consists of CNNSMC and graphic user interface (GUI), is being executed on a DSP. DSP is via analog outputs and digital inputs connected with servo-electronics, which is designed for the generation of the required current of DC driving motors and for measuring the position of robot axes with incremental encoders. On the other hand, the DSP-2 robotic

*Corr. Author's Address: University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, SI-2000 Maribor, Slovenia, jure.cas@uni-mb.si

controller via serial bus and by using library *ComVIEW* connected with the laboratory server, where the developed VI is running. By using the technology *Remote Panels* the laboratory server is also responsible for publishing the developed VI on web and for serving the remote client of experiment. Hardware implementation of experiment is shown in Figure 1.

1 SYSTEM COMPONENTS

The *DSP-2 Library for Simulink* [1] is a *MATLAB/Simulink* add-on toolbox, which provides rapid control prototyping support for DSP-2 robotic controller. The library contains a set of device driver's blocks for all available I/O ports of the DSP-2 robotic controller, including blocks for analog I/O, digital I/O, incremental encoder and blocks for communication between the PC and DSP-2 robotic controller (TCP/IP, USB in RS232). *Simulink Real Time Workshop* and the *DSP-2 Library for Simulink* enable developers to model applications in the *Simulink* block-diagram environment and, after successful simulation, quickly verify the designed algorithm with the DSP controller or DSP-2 robotic controller on a real device.

The *LabVIEW* virtual instrument named *ComVIEW* has been used for serving fly data visualization and parameter tuning tasks for the DSP-2 robotic controller. When the DSP target is selected in the *Simulink* model, a *LabVIEW* virtual instrument is automatically created from the *ComVIEW* template VI during binary code generation. A *ComVIEW* template contains an empty front panel and a fully functional block diagram. The block diagram implements functions for VI initialization, executable code download to the DSP-2 robotic controller, functions for transmitting and receiving messages between the PC and the DSP-2 robotic controller. During VI creation, numerical controls and indicators are automatically added to the VI front panel template, where the number of controls and indicators depends on the number of DSP communication blocks used in the *Simulink* model. Links between DSP signals and VI front panel objects are established programmatically using the *DSP Connection Manager Window*. This window appears on the PC immediately after the downloadable binary code starts executing on the DSP target system. Using mouse clicks, the user can create links between the VI front panel indicators and the DSP output signals, and links between the VI front panel controls and the DSP input signals or DSP parameters. When these links are set, a communication link is established between the VI running on the PC and the code being executed on a DSP controller. Whenever the controls on the VI front panel are changed, *LabVIEW* automatically downloads them via serial port, to the DSP controller. At the same time, all arrived DSP output signals are read from the PC serial port and displayed on VI control panel as graphs numerical indicators. In the *scope* mode, a small portion of code running on the DSP controller handles data acquisition and storage management. The selected DSP signals are, firstly, captured and then stored in the temporary controller's memory. After that, the captured data is transferred to the PC.

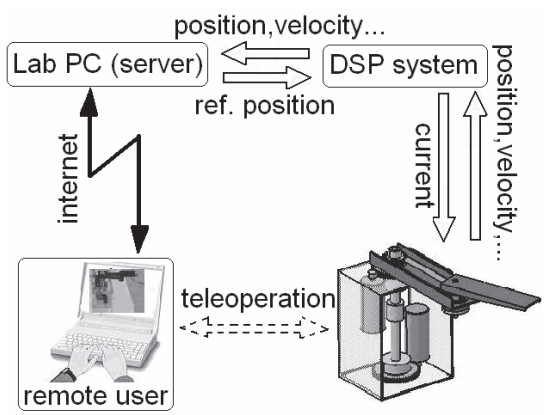

Fig. 1. Schematic implementation of the developed experiment for teleoperation of the SCARA via internet

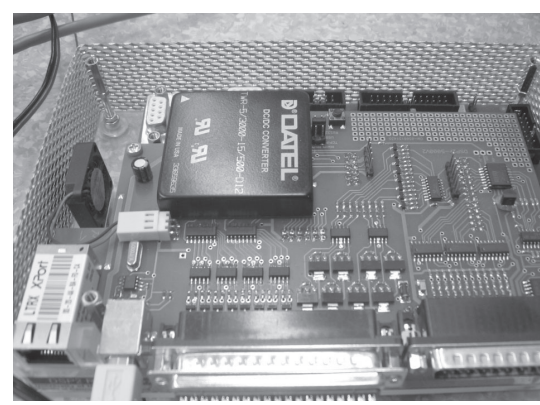

Fig. 2. On the DSP-2 robotic controller the developed software is being executed

Remote Panels is a *LabVIEW* add-on toolkit developed by National Instruments that enables the viewing and controlling of *LabVIEW* VI's over the Internet. Using this toolkit, the *LabVIEW* VI can be published on the internet with no additional programming. Afterwards, the virtual instrument can be remotely observed or controlled by using standard web browser. The remote user can fully access the user interface that appears on the web browser and, consecutively, has complete control of the remote application. Other users can point their web browser to the same URL to view a remote experiment. To avoid confusion, only one client can control the application at a time.

ComVIEW VI and the *LabVIEW* server are run on the same lab server. *ComVIEW* VI performs communication between the lab PC and the DSP-2 robotic controller, while the *LabVIEW* server enables remote operation of the *ComVIEW* VI. User of telerobotics application must have a *LabVIEW Run-Time Engine* installed on *home* computer in order to perform remote experiments.

The DSP-2 robotic controller [2] (Fig. 2) is composed of a DSP-2 controller and a DSP-2 add-on robotic board. The key components of the DSP-2 controller are the floating point digital signal processor, used for control algorithm execution, and the *Xilinx FPGA*, which implements peripheral interfaces. The DSP-2 robotic controller contains all the necessary peripheral for 4-axes robot control i.e. this system has 16 digital inputs, 8 digital outputs, 4 analog inputs/outputs, and 4 incremental encoder interfaces.

3 DESCRIPTION OF THE SCARA

The SCARA (Fig. 3) has two links. Position of the first link in the joint space is presented as the first degree of freedom (θ_1), and position of the second link in the joint space is presented as the second degree of freedom (θ_2) and θ is the vector of the real positions.

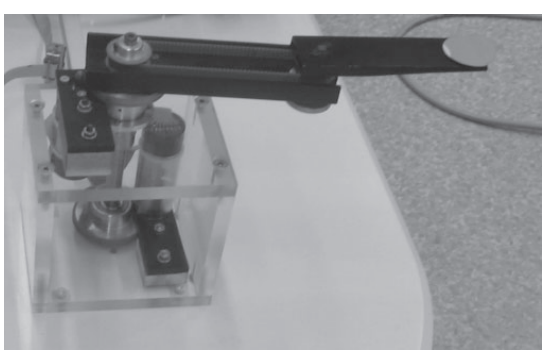

Fig. 3. The SCARA is shown

Each link is driven by a DC motor, and a gear-box with a transmission ratio 173/19. By using gear-boxes, any nonlinear inertia influences of the presented robot mechanism are decreased, but not completely. Additional dynamic nonlinearities are brought to the system as friction, which is proportionally large (about 20% of maximum torque) in the used SCARA.

Robot links are driven by a DC motor with nominal torque 28.4 [mNm]. The sinus signal from the incremental encoder is by servo electronics transmitted to 400 pulses per joint rotation.

A dynamic model of the SCARA mechanism, with two degrees of freedom is described by the Lagrange equation of motion, as follows:

$$T = M(\theta)\,\ddot{\theta} + h(\theta,\dot{\theta}) + F(\dot{\theta}) + T_n \qquad (1),$$

where T is a vector of the drive torque on the robot's joints, M is an inertial matrix, h is a torque vector due to the centrifugal forces, centripetal forces, and Coriollis forces, F is a torque vector due to friction forces and T_n stands for a torque vector due to unknown disturbances, $\dot{\theta}$ and $\ddot{\theta}$ are vectors of real velocities and accelerations of the robot joints. This well known mathematical note of a robot mechanism dynamics (1) can be expressed with an n-dimensional state-space system of equations with regard to the control value u:

$$\dot{x} = f(x,t) + B(x,t)\,u + d(x,t) \qquad (2).$$

New terms are defined as:

$$x \in \Re^n,\ u \in \Re^m,\ B(x,t) = \tilde{B}(x,t) + \Delta B(x,t) \quad (3),$$

where d is an unknown disturbance, B is an actual input matrix, $\tilde{B}$ is an estimated input matrix, u is a control vector, x is a state space vector of mechanism, and t stands for time.

4 DEVELOPMENT OF CNNSMC

The main advantages of the sliding mode control are the robustness to parameter uncertainty, loading disturbance, and fast dynamics response. However, these properties are valid on the sliding surface under the conditions

of modelling imprecision absence, external disturbances, and switching time delays. The control law of the sliding mode technique contains a discontinuous component and may, on the sliding surface, excite those high frequency dynamics neglected in modelling, due to high control activity. Practical sliding mode control implementations exhibit high frequency oscillations in the plant output, called chattering. This cause harmful effects such as torque pulsation in robot electrical drive and, consequently, imprecise positional control of a robot tip. Various methods have been proposed to eliminate chattering. The most commonly cited approach is smooth approximation of the switching element by saturation, so that a narrow boundary layer is introduced near the sliding surface. The control law is constituted by two components: a continuous law derived from the plant model using the Lyapunov theory, and a saturation law [3]. The first component is required to slide-down on the sliding surface, while the saturation law controls the unknown parts of the robot dynamics.

A number of researchers have suggested methods for alleviating the chattering effects and improve precision in the sliding mode position control of robot mechanisms. In [4] are developed perturbation estimation schemes; other authors suggested supplementing the control input with a predictive correction term [5]. Recently, soft computing techniques fuzzy logic [6], neural network [7] and genetic algorithm [8] have been used to estimate the robot non-linear dynamics, thus needing only the continuous law derived from the robot dynamics using the Lyapunov theory.

Our approach uses a neural network as an estimator for a part or, even complete, robot dynamics. We decided to use a neural network because of its high convergence speed in robot dynamics estimation, robust sliding-mode control scheme, and as little preliminary knowledge on the estimated mechanism dynamics model as possible. In our case, only nominal (average) values were used for inertia matrix parameters. Differences between actual inertia matrix parameters and nominal inertia matrix parameters represent structured uncertainties. Torque terms due to Coriollis forces and friction forces were neglected and they represent non-structured uncertainties. If we did not have a robust control scheme, the robot behaviour would be unpredictable during the first few moments of learning.

A well-known mathematical note of robot mechanism dynamics (Eq. 1) is transformed into an *n*-dimensional state-space system of equations with regard to the control value *u* (Eq. 2), because the theory of Lyapunov for searching the control law can only be used in the following way.

Our goal is to prove the stability of function for the robot system. This means that after transient time, defined with parameters of the matrix *G*, the difference between the actual and the desired vector of state, space variables *x* and x_r will equal zero and will be stable for all disturbances. Function $\sigma(x,t)=0$ will be stable if the Lyapunov function $V>0$ and the first Lyapunov time derivative of function is $\dot{V}<0$. According to the following definition:

$$\sigma(x,t)=G(x(t)-x_r(t))=G(x-x_r) \qquad (4),$$

where x_r and x are the vectors of the desired and actual state space variables and *G* is the matrix defining the control of system dynamics, we cannot prove the robot system's stability (Eq. 2). Nevertheless, we can look for suitable conditions for control law *u*, where the robot system will be stable. This is done in the following way.

For the simplest Lyapunov function *V*, to determine the control law *u*, the Equation (5) has been selected:

$$V=\sigma^T\sigma/2 \qquad (5),$$

The following equation is derived from(5):

$$\dot{V}=\sigma^T\dot{\sigma} \qquad (6).$$

Owing to the fact that $\dot{V}$ is not always less than zero for all x_r, *x*, and G, the first desired Lyapunov negative time function derivative is defined as:

$$\dot{V}=-\sigma^T D\sigma \qquad (7),$$

where *D* is a diagonal matrix with positive diagonal elements. If the Equation (7) and the derivative of Lyapunov's function (Eq. 6) are made equal, the result is:

$$\sigma^T(D\sigma+\dot{\sigma})=0 \qquad (8).$$

The Equation (8) is valid if both, or at least one, of the multiplicators equals zero. Since the first multiplicator, the term σ^T does not equal zero,

the control law can be calculated on the basis of the second multiplicator:

$$D\sigma + \dot{\sigma} = 0 \quad (9).$$

If the Equation (4) is differentiated and the Equation (2) is inserted into the recently calculated derivative, we obtain the following result:

$$\dot{\sigma} = G(f + \tilde{B}u + \Delta Bu + d - \dot{x}_r) \quad (10).$$

After the Equation (10) has been inserted into the implementation condition of control law (Eq.(9)), the result is as follows:

$$u = -(G\tilde{B})^{-1}\left[G(f + \Delta Bu + d - \dot{x}_r) + D\sigma\right] (11).$$

Since the term $(f + \Delta B \cdot u + d)$ is unknown and not measurable it is, therefore, approximated with the neural network $N = [o_1 \ldots o_i]^T$ by changing Equation (11) into:

$$u = -(G\tilde{B})^{-1}\left[G(N - \dot{x}_r) + D\sigma\right] \quad (12).$$

Since the term $(f + \Delta Bu + d)$ is unknown and non-measurable, a classic supervised weight learning of the neural network can not be used. Therefore, a so-called *on-line estimator* has been developed for estimating a learning signal (the difference between the target and the output of a neural network). The result after Equation (4) has been differentiated, is as follows:

$$\dot{x} = G^{-1}\dot{\sigma} + \dot{x}_r \quad (13).$$

After the Equations (12) and (13) have been inserted into the basic equation of mechanism dynamics, the result is as follows:

$$\dot{\sigma} + D\sigma = G(f + \Delta Bu + d) - GN = G(Z - N) (14),$$

where is substituted $Z = f + \Delta Bu + d$.

By using the derivative of Lyapunov's function and the Equation (14), the condition, where the system controlled by the developed control law (Eq. 12) remains stable, is assured:

$$\dot{V} = \sigma^T\dot{\sigma} = \sigma^T G(f + \Delta Bu + d - N) - \sigma^T D\sigma < 0 \ (15).$$

The next condition (16) has been developed from Equations (14) and (15). To make $\dot{V} < 0$ and consequently $\sigma \to 0$ possible, the condition expressed in Equation (16) has to be fulfilled during the time during in which the neural network is approximating the unknown part of robot dynamics $(f + \Delta Bu + d)$:

$$\left|G(f + \Delta Bu + d) - GN\right| = \left|D\sigma + \dot{\sigma}\right| < \left|D\sigma\right| \quad (16).$$

To learn about the output layer of neural network with two layers, a modified BPG rule has been used:

$$net_i = \sum_j w_{ij}o_j + b_i,\ o_i = g(net_i)$$

$$g(net) = 1 - 2/(1 + e^{-net}),\ \Delta w_{ij} = \varepsilon\partial E/\partial w_{ij} \quad (17),$$

$$E = (D\sigma + \dot{\sigma})^T(D\sigma + \dot{\sigma})/2 = (G(Z - N)^T)(G(Z - N))/2$$

$$\Delta w_{ij} = \varepsilon\partial E/\partial net_i \partial net_i/\partial w_{ij} = \varepsilon\partial E/\partial net_i o_j =$$

$$= \varepsilon\partial E/\partial o_i/\partial o_i/\partial net_i o_j \Rightarrow \Delta w_{ij} = \varepsilon\partial E/\partial o_i g'(net_i)o_j$$

where:

$$\partial E/\partial o_i = \partial/\partial o_i\left[(GZ - GN)^T(GZ - GN)\right]/2 =$$

$$= \partial/\partial o_i\left[(GN)^T(GZ - GN)\right] \Rightarrow \quad (18).$$

$$\Rightarrow \partial E/\partial o_i = \partial/\partial o_i\left[(GN)\right]^T(D\sigma + \dot{\sigma})$$

To learn the hidden layer of a neural network the traditionally back propagation rule is used.

From now, the detailed equations of control law for a SCARA mechanism with two degrees of freedom will be derived:

$$T = M\dot{\theta} + h + G_f + T_n \quad (19),$$

where T, h, G_f and T_n are column vectors of the 2 by 1 dimension, M is the matrix of the 2 by 2 dimension, and $\theta = [\theta_1\ \theta_2]^T$ is the column vector of the 2 by 1 dimension of two axes of the SCARA. The previous equation can also be rewritten in the following form:

$$\dot{x} = f(x,t) + B(x,t).u + d(x,t) \quad (20),$$

where:

$$x = \begin{bmatrix}\theta_1 & \theta_2 & \dot{\theta}_1 & \dot{\theta}_2\end{bmatrix}^T,$$

$$\dot{x} = \begin{bmatrix}\dot{\theta}_1 & \dot{\theta}_2 & \ddot{\theta}_1 & \ddot{\theta}_2\end{bmatrix}^T,$$

$$f = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \hat{M}^{-1}\left[\hat{h} + \hat{G}_f + \hat{F} + \hat{T}_n\right] \end{bmatrix}, \quad (21)$$

$$\tilde{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \multicolumn{2}{c}{\hat{M}^{-1}} \end{bmatrix}$$

and where $\hat{M}$, $\hat{h}$, $\hat{G}_f$ and $\hat{T}_n$ are estimated values of M, h, G_f and T_n.

Only nominal or average parameters are selected for the matrix $\hat{M}$. This means that all 4 parameters of the matrix are constant while the robot hand is moving. This is, of course, only a rough simplification of how things really look; for

it is a common fact that the parameters of the matrix vary according to individual axis movements in a robot's working space. Because of this, the unknown variable part exists and is estimated by the neural network (Eq. 12). The dimension of vector f is 4x1 and the dimension of the matrix is 4x2. The control law u of the 2x1 dimension is illustrated in the following equation:

$$u = -(G\tilde{B})^{-1}\left[G(N - \dot{x}_r) + D\sigma\right] \qquad (22),$$

where the variables are defined as:

$$G = \begin{bmatrix} K_{P1} & 0 & K_{V1} & 0 \\ 0 & K_{P2} & 0 & K_{V2} \end{bmatrix},$$
$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix},$$
$$x_r = \begin{bmatrix} \theta_{1r} & \theta_{2r} & \dot{\theta}_{1r} & \dot{\theta}_{2r} \end{bmatrix}^T, \qquad (23).$$
$$\dot{x}_r = \begin{bmatrix} \dot{\theta}_{1r} & \dot{\theta}_{2r} & \ddot{\theta}_{1r} & \ddot{\theta}_{2r} \end{bmatrix}^T \text{ and}$$
$$\sigma = G(x - x_r)$$

Coefficients of the matrices G in D should be selected in such a way that they enable the fastest convergence of neural network algorithm possible.

The column vector N is of the 4x1 dimension and represents the outputs of the neural network with i=1-4. The learning procedure for all the weights of an output layer is:

$$\begin{aligned} \Delta w_{1j} &= \varepsilon_J \begin{bmatrix} K_{P1} & 0 \end{bmatrix} (D\sigma + \dot{\sigma}) g'(net_1) o_j \\ \Delta w_{2j} &= \varepsilon_J \begin{bmatrix} 0 & K_{P2} \end{bmatrix} (D\sigma + \dot{\sigma}) g'(net_2) o_j \\ \Delta w_{3j} &= \varepsilon_J \begin{bmatrix} K_{V1} & 0 \end{bmatrix} (D\sigma + \dot{\sigma}) g'(net_3) o_j \\ \Delta w_{4j} &= \varepsilon_J \begin{bmatrix} 0 & K_{V2} \end{bmatrix} (D\sigma + \dot{\sigma}) g'(net_4) o_j \end{aligned} \qquad (24),$$

where:

$$net_i = \sum_j w_{ij} o_j + b_i \qquad (25)$$

and where j = 1...20, I = 1...6, l = 1...4, and g' = 1 is the first derivative of the output function.

The neural network inputs are: two actual positions, two actual velocities and two differences between the reference and the actual position in joint coordinates.

Figures 6 and 7 show, that the developed CNNSMC is appropriate as a control scheme for the robot control. In the Figure 6, the positional error for the positional reference values in joint space is shown. The reference value is described as follows:

$$\theta_{1\_ref} = \theta_{2\_ref} = 3\sin(0.5 \cdot t) \qquad (26).$$

At the beginning of test experiment the positional error is proportionally huge. The reason is a random value of the neural network weights at the beginning of experiment. After first few moments neural network becomes stable and the positional error during movement of robot is approximately 5 mm. After 30 seconds the reference values in joint space (θ_{1_ref}, θ_{2_ref}) are constant. Measured static error after 30 seconds is equal to zero.

For measurement of sum square positional error (SSEp) of the robot in Cartesian space, the Equation (27) is used. The result (Fig. 6) shows the convergence ability of a CNNSMC for the above described experiment.

$$SSE_P = \sum_{i=1}^{2000} ((X_{ref\_i} - X_i)^2 + (Y_{ref\_i} - Y_i)^2) \qquad (27),$$

where X_{ref_i} and Y_{ref_i} are reference coordinates in Cartesian space, X_i and Y_i are actual coordinates in Cartesian space and i is the number of measurements. It is determined with the equation:

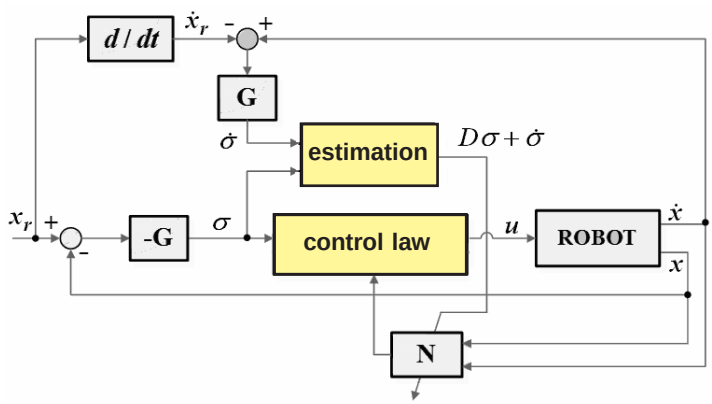

Fig. 4. Control scheme of CNNSMC for SCARA teleoperation is shown

$$i = \frac{T_p}{T_s} = \frac{2s}{0.001s} = 2000 \quad (28),$$

where T_s is the sample time and T_p is the time period of measurement.

5 IMPLEMENTATION OF EXPERIMENT

Telerobotic application is designed for educational purposes of students. Only predefined conditions for using it are reliable internet connection and installed *LabVIEW*'s free library *LV Run Time Engine.* When a user is intending to take a control over the telerobotic application, the internet address of application is written to the internet browser program. The falling menu appears using the right-mouse click. There the option *Request control of VI* must be selected.

After control is established, the dialog window appears. The user name and user e-mail address must be written there, because the results of the experiment are sent via an *e-mail*. When the button *SEND E-MAIL* on the control panel is chosen, the results are sent to a defined *e-mail* address. The results are attached in folder *rezultati.zip*, which consists of:

rezultati.rtf the front panel and block diagram of the used virtual instrument is attached in this folder.

rezultati.dat the text-data of all measured signals is contained in this folder (in column form). The first column is time data, and the other columns contain the measured data. This format can easily be imported to *MATLAB*, where it can be observed in graph form.

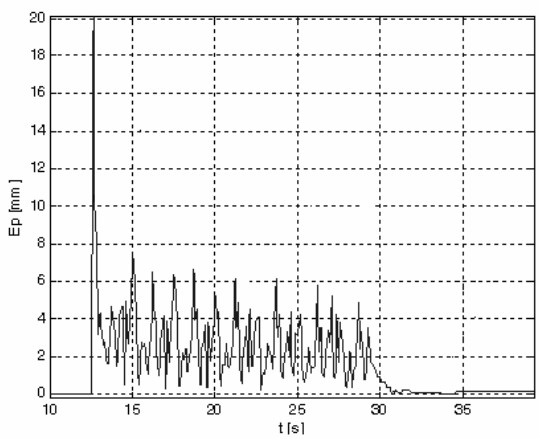

Fig. 5. Measured dynamic error for reference position movement (Eq. 26) and measured static error (after the 30s period of experiment is elapsed) of the SCARA's tip

After the initial procedure of a web-based educational tool is completed, the front panel of the virtual instrument appears (Fig. 7). By using the switch on the top left rectangle of the control panel, it is possible to choose from three different running modes of the SCARA. The holding robot in the fixed-point is the first running mode, rotating robot links with sine reference in a joint-space is the second running mode, and defining the top coordinates of the robot arm in the Cartesian space by mouse clicking on the working area, which is presented as the grid on control panel, is the third running mode.

When the second running mode of the robot mechanism is chosen, the user is able to change the rotation frequency of the SCARA's axes from zero to half-radian per second in joint space and the amplitude of rotation from zero to *pi* radians and two graphs appear on the control panel of the virtual instrument. The referenced and actual position values of the first robot axis are shown on the upper graph, and the referenced and actual position values of the second axis are shown on the bottom graph. In addition, a positional error of robot, the SSEp in the joint space and the Cartesian space, values of some weights of neural network and the first output from neural network, can also be shown on both graphs.

In the middle of the control panel, within the small rectangles, the numerical values of the robot's position in the joint space, the values of some neural network's weights, the value of the first output from the neural network, the values of

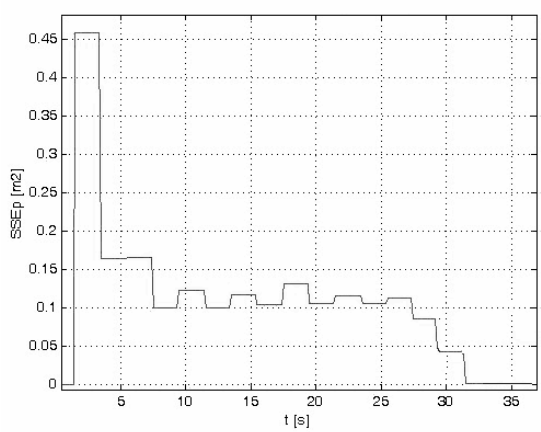

Fig. 6. The SSEp of SCARA's tip for the reference position movement (Eq. 26)

sum square errors in the joint space and Cartesian space, and the positional error value of the robot's top in Cartesian space, can be observed in the same control panel.

By using the numerical controls on the left middle rectangle on the control panel, the user can change the values of CNNSMC parameters at predefined intervals. The performance of CNNSMC changes when changing these parameters. The work is to find the best parameters, which would ensures that the performance of CNNSMC is optimal. Values of parameters are shown in Table 1.

6 HYPOTHETICAL USE OF THE DESCRIBED EXPERIMENT IN REALISCT MANUFACTURING ENVIRONMENT

The research field of production technology is often focused in the development of intelligent distributed production plants. The intelligent production plant is a machine, a part or whole production plant, which is able to self-adapt of modifications of the production process and environment, which has also influence on the production process. By involving the optimization algorithms (e.g. genetic algorithm), the intelligent production plant is able to optimize the production process as well.

The developed experiment for teleoperating of the SCARA by CNNSMC is possible to implement in realistic manufacturing environment as a part of intelligent distributed production plant. The self-adapting ability of CNNSMR and possibility of teleoperation the experiment via the internet connection are the qualities, which classify the developed experiment in the class of the intelligent production plant.

If the control system dynamics of hypothetic production process is changed because of the load modifications and the variation of the friction forces value in bearings (usage and environment temperature), the developed CNNSMC is able to self-adapt on the dynamics changes. But, when the

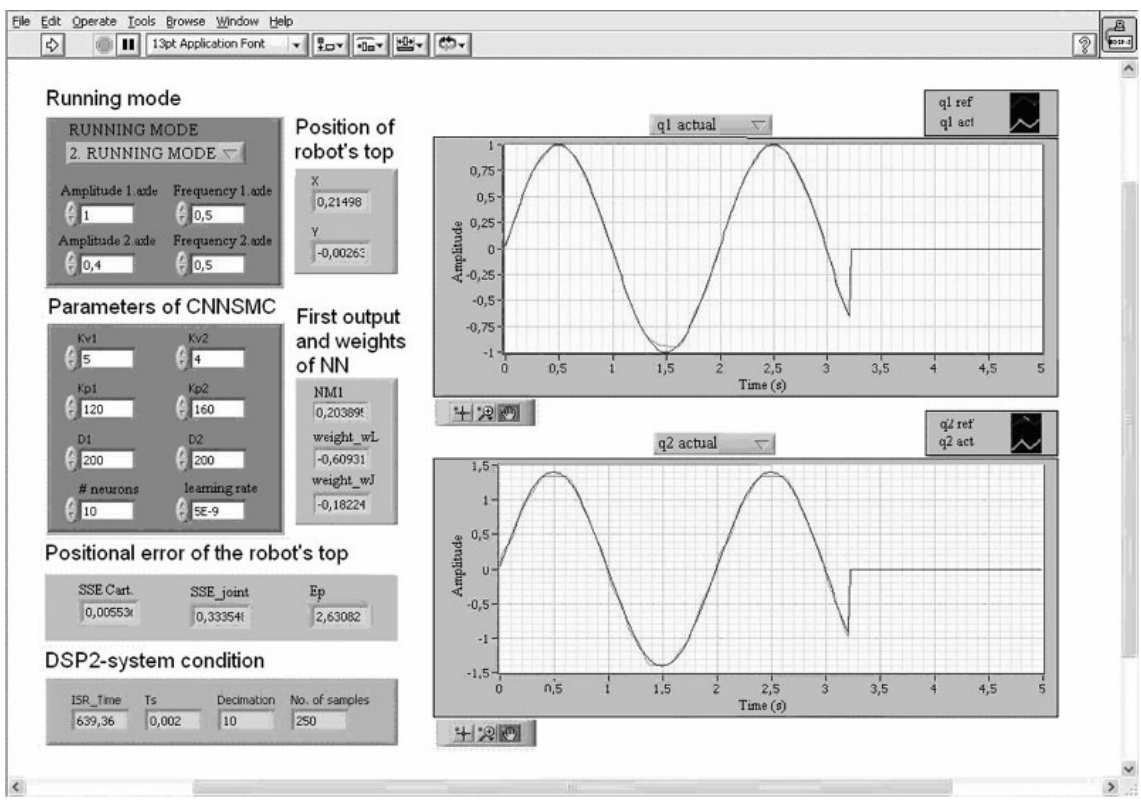

Fig. 7. By using the standard Internet browser, the remote user is able to access the control panel of experiment

Table 1. *The remote user of experiment is able to set the values of CNNSMC's parameters within the predefined intervals*

Symbol of parameter	***Value of parameter interval***	***Short description of parameter***
K_{V1}	*[2, 20]*	*velocity coefficient (1st degree of freedom)*
K_{V2}	*[2, 20]*	*velocity coefficient (2nd degree of freedom)*
K_{P1}	*[20, 200]*	*positional coefficient (1st degree of freedom)*
K_{P2}	*[20, 200]*	*positional coefficient (2nd degree of freedom)*
D_1	*[20, 400]*	*dynamic coefficient (1st degree of freedom)*
D_2	*[20, 400]*	*dynamic coefficient (2nd degree of freedom)*
nr	*[2, 20]*	*number of neurons in hidden layer*
eta	$1 \cdot 10^{-8}$	*learning rate*

system dynamics is changed even more, the self-adapting ability of CNNSMC can not optimize the regulation process. In that case, the remote user is able to fix the parameters of proposed controller to optimize the performance of CNNSMC again. The value of the reference position and speed of the production plant degrees of freedom is possible to set by teleoperation process as well.

7 CONCLUSION

The paper shows the intelligent interface, based on the neural network control approach for adaptation to environment disturbances (e.g. friction etc.) for telerobot application. The intelligent interface is used as a web-based educational tool for teaching neural network robot control techniques in telerobotics. When the robot dynamic is changed, this information is unknown to the user and because of this he is unable to dispatch it manually. This is not a problem for control however, because when CNNSMC is used, the approximation of changed robot dynamics is computed independently of the remote's user. The presented application's additional advantage is the possibility of on-line tuning parameters using CNNSMC, with the numerical controls of GUI, thus improving the performance of the robot's control and also making application more appropriate as an educational tool for students. With the possibility of on-line tuning parameters and observing responses of control using graphs and numerical indicators, students are able to learn how the parameters of CNNSMC affect performance.

8 REFERENCES

[1] Hercog, D., Čurkovič, M., Edelbaher, G., Urlep, E. Programming of the DSP2 board with the MATLAB/Simulink. *Proceedings IEEE ICIT 2003*, December 2003, p. 709-713.

[2] DSP-2 web page: www.ro.feri.uni-mb.si/projekti/dsp2.

[3] Slotine, J., Li, W. *Applied nonlinear control*. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[4] Curk B., Jezernik K. Sliding mode control with perturbation estimation: application on DD robot mechanism. *Robotica*, vol. 19, 2001, p. 641-648.

[5] Kaynak, O., Denker, A. Discrete time sliding mode control in the presence of system uncertainty. Int. J. Control, vol. 11, 1993, no. 7, p. 665- 678.

[6] Rojko, A., Jezernik, K. Sliding-mode motion controller with adaptive fuzzy disturbance estimation. *IEEE trans. ind. electron.*, vol. 51, 2004, no. 5, p. 963-971.

[7] Šafarič, R., Jezernik, K., Pec, M. Neural network control for direct-drive robot mechanisms. *Engineering application of artificial intelligence*, vol. 11, 1998, no. 6, p. 735-745.

[8] Fujisawa, S., Obika, M., Yamamoto, T., Kawada, K., Sueda, O. Speed control of 3-mass system with sliding mode control and CMAC. IEEE *International Conference on Systems, Man and Cybernetics*, vol. 5, 2004, p. 4400-4407.