BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 68, No. 5, 2020
DOI: 10.24425/bpasts.2020.134667

Adaptive controller design for electric drive with variable parameters by Reinforcement Learning method

T. PAJCHROWSKI*, P. SIWEK, and A. WÓJCIK

Poznan University of Technology, Institute of Robotics and Machine Intelligence, Piotrowo 3A, 60-965 Poznań

Abstract. The paper presents a method for designing a neural speed controller with use of Reinforcement Learning method. The controlled object is an electric drive with a synchronous motor with permanent magnets, having a complex mechanical structure and changeable parameters. Several research cases of the control system with a neural controller are presented, focusing on the change of object parameters. Also, the influence of the system critic behaviour is researched, where the critic is a function of control error and energy cost. It ensures long term performance stability without the need of switching off the adaptation algorithm. Numerous simulation tests were carried out and confirmed on a real stand.

Key words: Reinforcement Learning, adaptive control, electric drive, machine learning.

1. Introduction

In recent years, an increasing demand for the quality of servo-drives is observed and its even further growth can be expected. Many of these controlled devices e.g. industrial robots, machine tools, wind turbines, paper machines or rolling machines have a complex mechanical structure [1–6]. The increasing requirements for control performance imply that the drive systems should be insusceptible to changes in parameters or should be able to adapt. This issue is important, especially in the so-called direct drives. The direct drive is a mechanical solution in which the electric motor is connected to the load without a gearbox. A special construction of the motor is required, that enables the drive system to operate with a low rotational speed, generally below 100 rpm, and often even less. Lack of mechanical transmission has many advantages, e.g.: elimination of backlash introduced by the transmission (which improves static accuracy of work and dynamic properties of the drive), increase in the efficiency of the drive system (elimination of losses in the transmission) and reliability (fewer mechanical components). The abovementioned advantages of direct drives require using appropriate algorithms in the control system. In a direct drive, the moment of inertia of a running machine is several times greater than the moment of inertia of the motor itself, therefore large changes in the moment of inertia in the machine affect the operating conditions of the drive system (in a classic drive with a gearbox, the moment of inertia converted to the motor has a similar value).

In many drive systems, the moment of inertia depends on the angle of the shaft position. Therefore, there arises a need for a position or speed control system with adaptive properties. It is not trivial to propose suitable control algorithms or structure, which would ensure high performance and control accuracy of an electric drive. This problem concerns many industrial and scientific institutions all over the world.

In order to ensure proper quality of the drive system operation under changing mechanical parameters of the drive, robust [4] or adaptive [5] controllers are used, which very often implement computational intelligence methods, especially artificial neural networks [6, 7]. The use of neural networks for speed control is widely presented in the literature [7–14]. Different types, learning algorithms and structures of neural networks have been implemented. Papers [8–10] present two different concepts of adaptive neural network controller. In [10] an adaptive controller with a reference model is presented, where the neural controller learns from the error between the output signal of the reference model and the actual signal of the controlled object. Paper [11] presents the concept of a neural control with feedback from the reference current. Another interesting concept of neural controller is given in [8], where the repetitive neural controller is used. In articles [12, 13] a model of an inverse object is applied and in [14] – an internal model control based on backpropagation neural network inverse system. The main disadvantages of this solution are problems with selecting proper teaching data, learning parameters and the structure of the neural controller.

From the cited literature we can deduce that nowadays we face an increasing interest in machine learning. One of the definitions of machine learning is presented in [15]. In general, to perform a task, the system learns from the delivered data set. A dataset is a set of numerous examples, where an example is a collection of attributes. Machine learning is fundamentally divided into three types: Supervised learning, Unsupervised learning, Reinforcement learning (see Fig. 1). The goal of supervised learning is to find an approximation of the function

*e-mail: Tomasz.Pajchrowski@put.poznan.pl

Manuscript submitted 2020-01-10, revised 2020-03-30, initially accepted for publication 2020-04-15, published in October 2020

that binds input and output data. Classification and regression are two of the most common supervised machine learning tasks. The main idea of unsupervised learning is to search for possible patterns and relationships in unclassified data. One common task of unsupervised learning is to group similar examples, which is called clustering. Reinforcement learning refers to goal-oriented algorithms that learn how to achieve a complex goal or maximize the outcome in a specific way in many steps. A simple reward feedback is required, i.e. the so-called reinforcement signal. Nowadays a lot of research is conducted pertaining artificial intelligence and one of its branches – the machine learning [16–18]. It is an interdisciplinary domain with particular emphasis on fields such as computer science, robotics and statistics. Particularly interesting research area pertaining the control of electric drives seems to be Reinforcement Learning (RL) [19–21]. Reinforcement Learning is a type of Machine Learning (see Fig. 1), and thereby also a branch of Artificial Intelligence. It allows machines and software agents to automatically determine an ideal behaviour within a specific context, in order to maximize their performance. It allows a machine or software agent to learn how to operate based on feedback from the environment. This action can be learned once and for all or adjusted over time. We are dealing with an automated learning method, which suggests that there is no need for a human expert to know a specific application or plant. In electrical drive applications, especially in robot drives where the moment of inertia and load torque change simultaneously and often as a function of angle, the use of RL seems justified. In [22, 23], authors attempt to design an adaptive control system using RL technique and neural network. Changes in the controller parameters are updated online based on information that comes from a predefined critic and the environment.

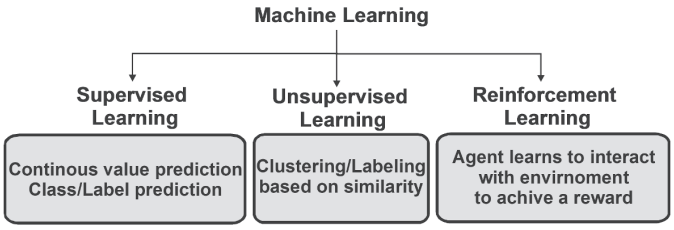


Fig. 1. Fields of machine learning [23]

The main purpose of this article is to keep the angular velocity constant, despite the variable moment of inertia or disturbance, and to emphasize the important issues concerning the design of neural controllers and their long-term performance stability, without the need of switching off the adaptation algorithm by the RL method.

An important novelty proposed in the article is the definition of a new critic as a function of error and control cost, which ensures the stability of the drive system in a long-term horizon. Numerous simulation studies have been carried out, where the parameters of the critic proposed in the article were modified to take a wide range of values. The proposed concept was successfully tested on a laboratory stand.

The paper is organized as follows. Section 2 presents the structure of the drive system and describes the laboratory stand. Section 3 focuses on the idea of RL control for the issue under consideration. Section 4 presents the proposed concept of RL control with neural networks. Sections 5 and 6 present simulation and real studies, respectively. Section 7 is devoted to the conclusions and future work.

2. PMSM drive system

2.1. The structure of the motor control. In presented article, a speed controller for a direct drive with a synchronous permanent magnet motor (PMSM) was researched. Block diagram of the drive with PMSM is shown in Fig. 2. In the paper, authors apply a Field Oriented Control of 3-phase motor using constant angle of power $\delta = \pi/2$, which corresponds to zero current in the d-axis.

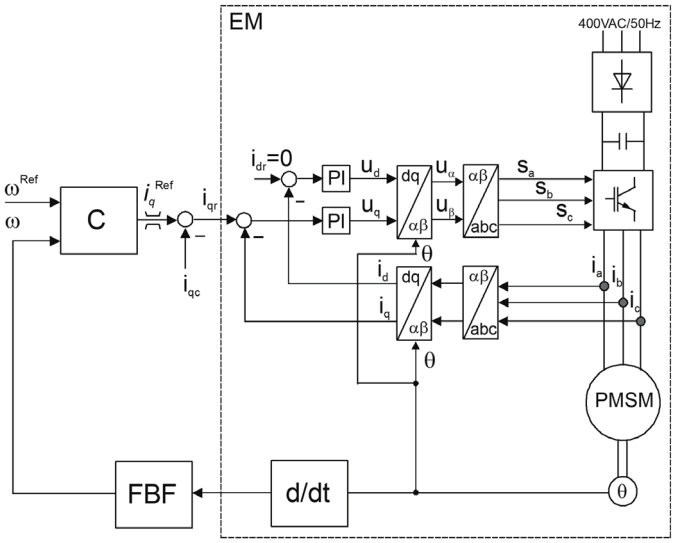


Fig. 2. The structure of the motor control PMSM. (*FBF* – anti-resonance filter, *EM* – part of the electromechanical system, *C*-speed controller, i_{qref} – controller output signal, i_{qc} – auxiliary input for compensation of periodic and friction torque, i_{qr} – q-axis reference current)

2.2. Laboratory Stand. To realize the research, a laboratory stand (Fig. 3) was used, with a changeable moment of inertia and the load torque, as function of angle of the shaft according to the equation:

$$T_d = J\frac{\mathrm{d}\omega}{\mathrm{d}t} + \frac{\omega^2}{2}\frac{\mathrm{d}J}{\mathrm{d}\theta} \tag{1}$$

where: J – moment of inertia, ω – angular speed, T_d – dynamic torque, θ – shaft position.

Figure 4 shows the solution of the Eq. (1), which was used in performed simulations. Figure 5 presents changeable moment of inertia as a function of the angle. The minimum and maximum value depends on the number of placed metal rings and the shaft angle. The minimal value of inertia is 1.09 kgm^2 (one

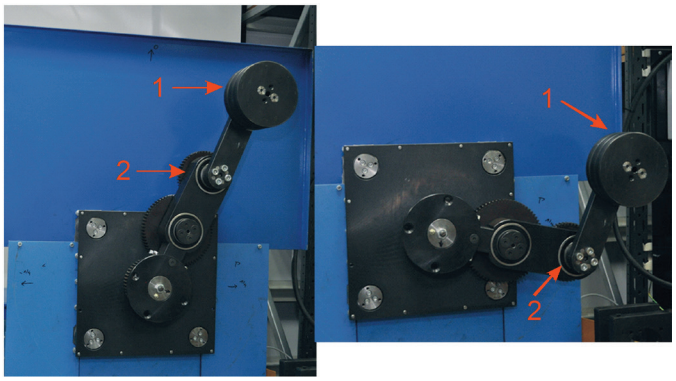


Fig. 3. Photography of the load continuously variable moment of inertia. 1 – metal rings for changing the weight; 2 – arm of variable length

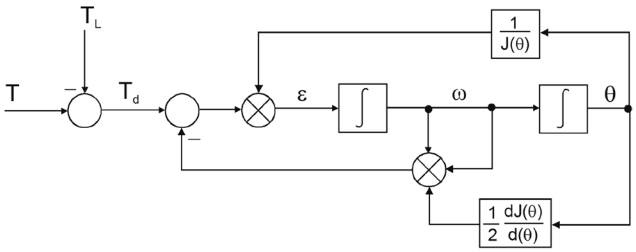
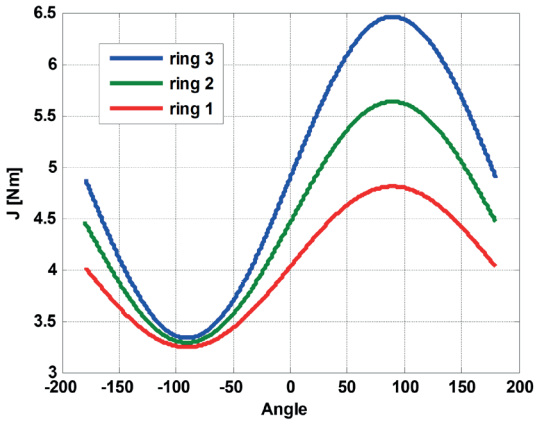


Fig. 4. A block diagram of the system with variable moment of inertia depends on the angle of rotation of the shaft



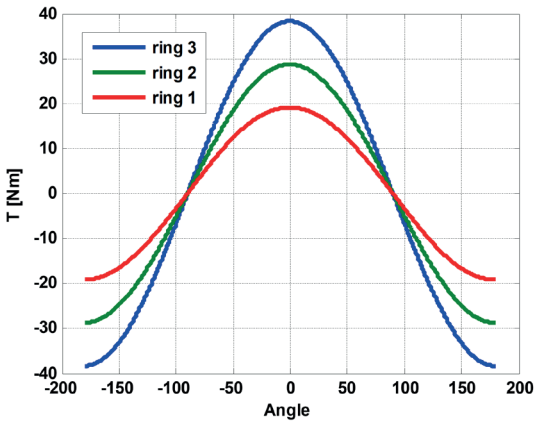


Fig. 5. Waveforms moment of inertia and load torque for the different combinations of the metal rings

metal ring located in the lowest part of the drive), and the maximum value is 3.77 kgm^2 (three metal rings located in the highest part of the drive). The biggest load torque is observed in positions 0 and 180 degree, as shown in Fig. 5.

3. Reinforcement Learning (RL) method

Reinforcement Learning (RL) is a type of machine learning technique (see Fig. 1) that enables an agent to learn in an interactive environment by trial and error, using feedback from its own actions and experiences [24]. Unsupervised and reinforcement learning (see Fig. 1) [24] use mapping between input and output, unlike supervised learning, where feedback provided to the agent is a set of correct actions required to realize the given task. Reinforcement learning, on the other hand, uses rewards and punishments as signals for positive and negative behavior. Compared to unsupervised learning, reinforcement learning is different in terms of goals. The goal in unsupervised learning is to find similarities and differences between data points, whereas in reinforcement learning the goal is to find a suitable action to maximize the rewards of the agent. Figure 6 shows the basic idea and elements involved in a reinforcement learning model.

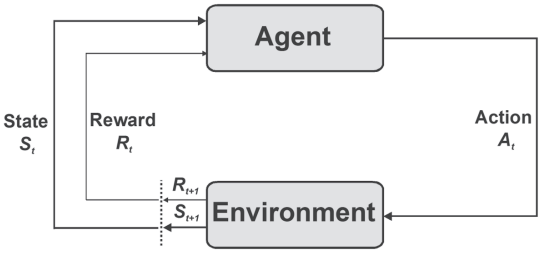


Fig. 6. Basic idea a reinforcement learning model [23]

The key terms that describe the elements of a RL problem are [24]: (1) Environment: the world which the Agent can interact with (2) Agent: a hypothetical entity which performs actions in an environment in order to get some reward, (3) State: the current state returned by the environment, (4) Action: all possible moves that the agent can perform, (5) Reward: an immediate feedback sent from the environment to evaluate the last action performed by the agent, (6) Policy/Critic: the strategy that the agent employs to determine next action based on the current state.

Several RL methods in machine learning have been developed and successfully applied to learn an optimal policy for Markov's decision-making processes (MDP – Markov Decision Processes) in real time, as shown in Fig. 7 [24]. An analogous control system using RL is shown in Fig. 8, where the controller, based on state feedback and reinforcement feedback about its previous action, calculates the next control signal which should lead to an improved performance.

The controller–critic structure is shown in Fig. 8 [25]. This structure allows the algorithm to calculate optimal decisions in real time, where the controller affects the environment, and the critic evaluates those actions. The learning mechanism sup-

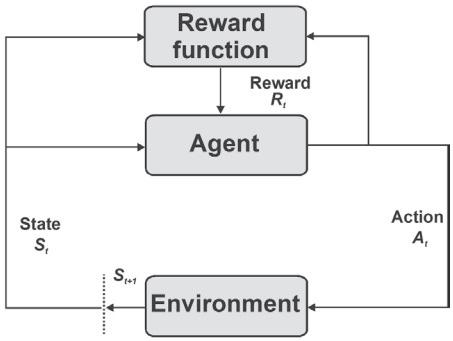


Fig. 7. Reinforcement Learning for MDP [23]

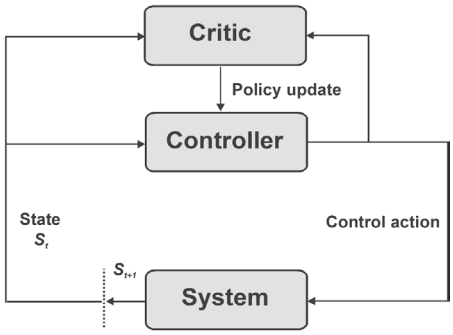


Fig. 8. Reinforcement learning with a controller–critic structure for control system [23]

ported by a controller-critic structure consists of two stages: policy evaluation by the critic and subsequent policy improvement by the controller. The policy evaluation phase is carried out by monitoring the environment behaviour and results of the current control activities. Obtained results are evaluated using a performance index. A policy can be defined in terms of optimality, such as minimum fuel consumption, minimum risk or maximum reward. In this system, reinforced learning is a way of learning about optimal behaviour by observing real-time responses from the environment under no optimal control policies.

4. Control by RL

In Fig. 9, the concept of reinforcement learning with a controller–critic structure for PMSM control system is presented. The structure consists of:

- Environment:
  - a mechanical part with a variable moment of inertia and a load torque as a function of the motor shaft position, described by formula (1),
  - closed torque control – if realized with well-tuned controllers, in speed control research it can be approximated by the following formula [26]:

$$G_T(s) = K_T \cdot e^{-sT_{\mu i}} \tag{2}$$

where: K_T – torque constant 17.5 Nm/A, $T_{\mu i}$ – total system measurement delays 0.5 ms,

- Agent:
  - critic – which evaluates the quality of the control and is described by the following formula:

$$f_{cr} = f\left(\omega^{\mathrm{Ref}}(n),\ \omega(n),\ i_q^{\mathrm{Ref}}(n)\right) \tag{3}$$

where: ω^{Ref} – reference speed, i_q^{Ref} – reference current in q axis,

  - controller – generates control signal in sample n, based on the reference value and previous measuring samples and according to the formula:

$$i_q^{\mathrm{Ref}}(n) = f\big(\omega^{\mathrm{Ref}}(n),\ \omega(n),\ \omega(n-1),\ldots \\ \ldots,\ \omega(n-k),\ \theta(n)\big). \tag{4}$$

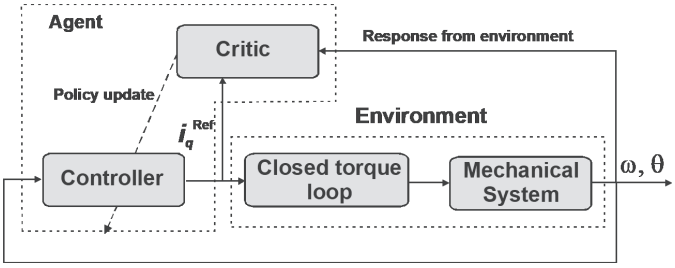


Fig. 9. Reinforcement learning with a controller–critic structure for PMSM control system

In the following research authors assumed neural network as the controller structure. The mentioned network is a feedforward network, consisting of three layers of sigmoid neurons. All its neurons have a hyperbolic tangent activation function.

4.1. Critic function. In controlling the motor speed of PMSMs, the angular speed error is commonly used to create controlling signals. Therefore, it is natural to try to optimize the neural network with a similar dependency. This kind of critic can be defined by the following formula:

$$f_{cr} = C_e\left(\omega^{\mathrm{Ref}} - \omega\right) \tag{5}$$

where: C_e – control error coefficient.

Using this kind of critic will result in a global minimization of the object response error. If the reference value step changes will be fed to the neural network, then one should expect that the moment right after the step change will have a higher weight than the small speed changes oscillating the set value. Even if using a reference model that generates reference signal adjusted to the time constants of the real object, because of the imperfectness of the reality description, small differences would still exist and result in obtaining an error. In case of significantly long-time horizons, the learning algorithm will cumulate the error and thus forcing an increasingly forceful response of the network, which will in consequence worsen the quality and cost of control.

In order to achieve a globally convergent process of controller's learning, there is a need to choose a function that takes into account the cost of the control, so that the controller will be punished for issuing a too high control signal throughout its whole performance. This publication uses the critic function in the following form:

$$f_{cr} = C_e\left(\omega^{\mathrm{Ref}} - \omega\right) - C_c i_q^{\mathrm{Ref}} \tag{6}$$

where: C_c – controller cost coefficient.

4.2. Neural network structure. In presented research two possible structures of neural controllers were analysed. First one (Fig. 10) has the classic structure of the feedforward type, which behaves in a way similar to the PD-type controller, whose main disadvantage is the fixed error.

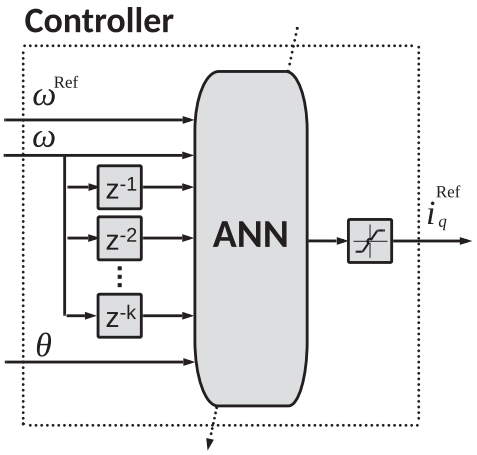


Fig. 10. Internal structure of the neural network-based controller

To eliminate the fixed error, the network structure was extended to integrate one of the network outputs after strengthening it by K_i fold. The obtained neural network is shown in Fig. 11.

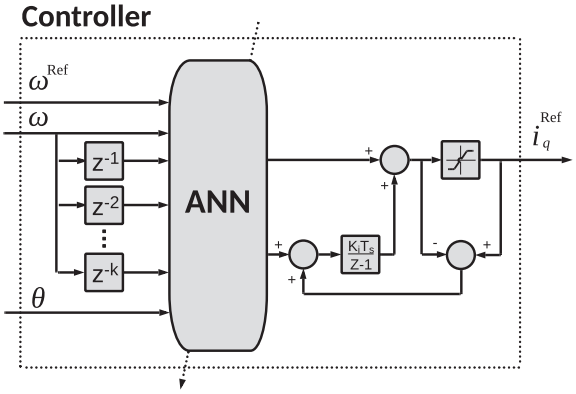


Fig. 11. Internal structure of the neural network-based controller with integration

Presented ANN has been taught with the use of an error backpropagation algorithm. Teaching neural networks with this algorithm is possible only if one can indicate the error gradient of the network. To calculate this gradient, knowledge about the teaching data is required, namely the knowledge about pairs consisting of neural network input vectors and output errors calculated for these particular vectors. In case of unknown network outputs, which is the case of control systems, error calculation is not trivial. Because of this, methods such as Model Reference Adaptive Control (MRAC) are used. This particular method is characterized by one of the networks being taught to reconstruct the object responses, while the second one works as a controller. Thanks to this, it is possible to calculate the control network error based on the backpropagated control error, which is obtained from the network used for modelling the object.

In order to solve this problem, in presented research, the authors applied an ANN taught by a RL algorithm. In RL, changes in the environment that are caused by agent's actions are observed, and result in the agent being rewarded or punished by a critic for the particular action in consideration, according to the formulas (5) or (6) respectively. The formulas numerically define the sign of the network error gradient, which is caused by the agent's actions. To calculate its approximate values, the neural network input vector is required, which is responsible for causing the measured changes in the environment. If the approximate values of the error gradient are known, it is possible to appoint changes in ANN weights, which will bring it closer to the optimal solution. The algorithm can be presented as in Fig. 12.

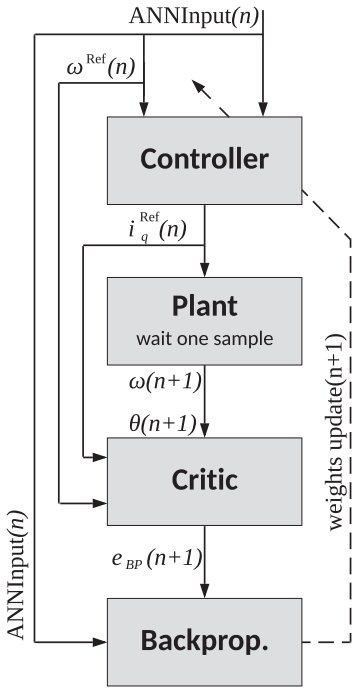


Fig. 12. Reinforcement Learning scheme for neural speed controller

5. Simulations results

Simulation research was performed in MATLAB Simulink 2019a environment, on the device model presented in Fig. 2. A continuous-time model of the drive system was used for the

simulation, using a variable-step solver. The drive model outputs were sampled with a constant step of 100 µs. Control system was implemented in the discrete form with sample time of 100 µs. The reference speed signal was generated based on random values from a uniform random number generator from the $[-0.5; 0.5]$ interval. Authors assumed that to precisely define the influence of a given parameter on the controller, all simulations should start form the same point and the controller should always be fed the same teaching data. To achieve this, each simulation was preceded by re-setting the generator seed to a single, fixed value. Thanks to such an approach, ANN began each simulation with the same weights, and the reference signal was the same for each test.

The following article focuses mostly on presenting the long-term correct system performance.

The first performed simulation test (Fig. 13) was to verify the operation of the speed control system, without integration, which uses the critic given in (5). Said critic is dependent only on the control error and has the form that can be easily found in cited subject literature. The analysis consisted of checking the initial learning phase of the regulator, which was fed pseudo-random step changes in the reference value. The controlled object, according to (1), was characterized by angle-dependent inertia and load torque.

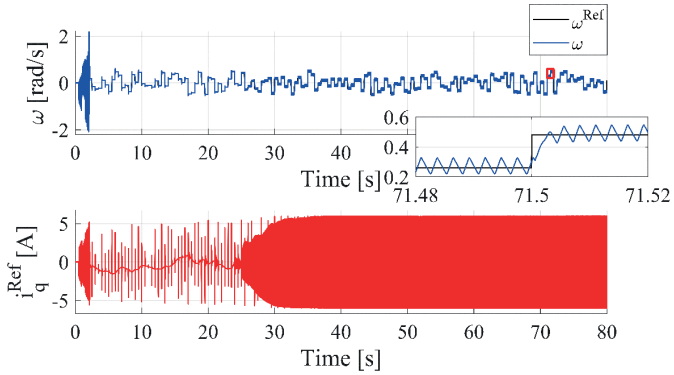


Fig. 13. Initial learning phase of the network for critic given by the relation (5)

The initial control phase is stable, the system adapts in less than 2 s to the controlled object and is characterized by no fixed error. However, after 23 s of simulations, the control system starts to oscillate in an unfading way. This kind of state holds independently of the simulation time. One can find solutions of this problem in subject literature, e.g. limiting the maximal values of weights or turning off the learning algorithm after some time. Yet, all the proposed methods have their disadvantages. Limiting the maximal values of weights reduces the space of possible results, which increases the chance of finding optimum being not possible. Moreover, there arises the problem of appointing the limit itself. On the other hand, turning off the learning after a set time, causes the algorithm to be unable to find solutions for control objects which slowly change their parameters.

Figure 14 presents the initial learning phase of a network, without integration, for the critic proposed in paper (6). The controller under consideration adapts significantly longer to the plant (42 s). During that time, weights of the neural network undergo small changes. After the initial adaptation time, the controller starts to visibly improve its performance, forcing the object to aperiodically change its state. Because in the considered research case the learning algorithm does not substitute the control error integration, a significant fixed error occurs.

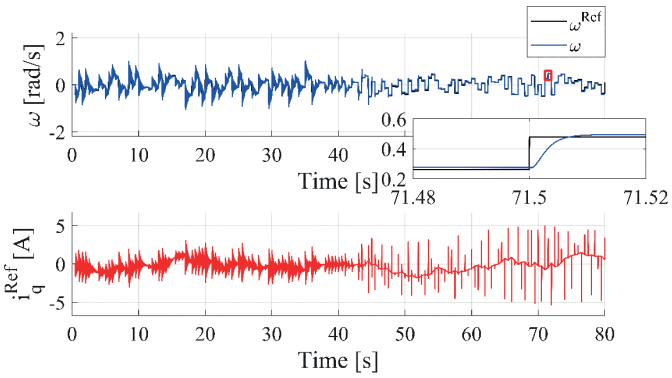


Fig. 14. Initial learning phase of the network for critic given by the relation (6)

Figure 15 presents the performance of the system after 900 s of learning. One can notice constant changes of T_L and J, which are reflected in the average value of the reference current. One can also see that the controller still works stable after this time.

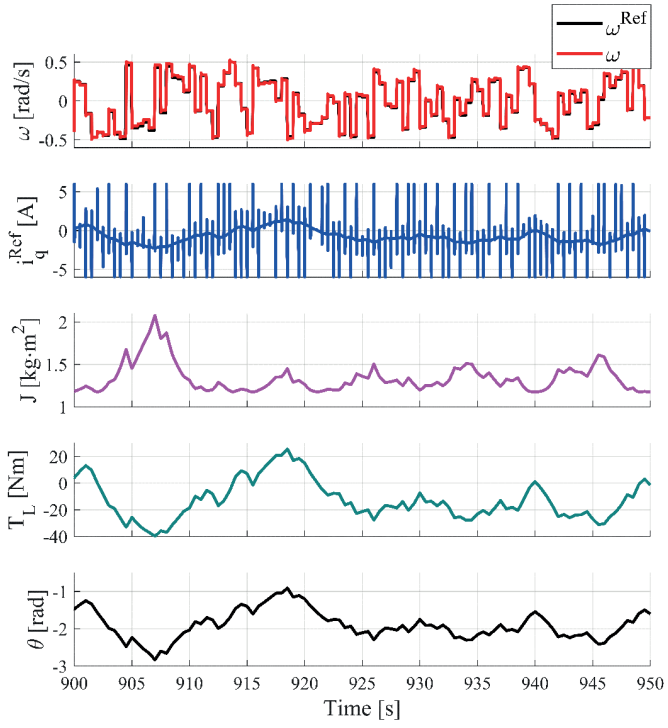


Fig. 15. Control system response after 900 s of learning

Critic presented in (6) is characterized by proportionality coefficients between the reference current in the q axis: C_c and

the control error: C_e. The proportion between those parameters sets the impact of the speed control reaction to the changes in the reference value or disturbances occurring in the plant. The smaller the C_c, the higher the cost of control expressed by the Integral Absolute Cost indicator (7):

$$IAC = \int_0^\infty \text{abs}\left(i_q^{\text{Ref}}\right) \text{d}t. \tag{7}$$

The influence of C_c is shown in Fig. 16. One can notice that the neural controller behaves according to the formula stated above.

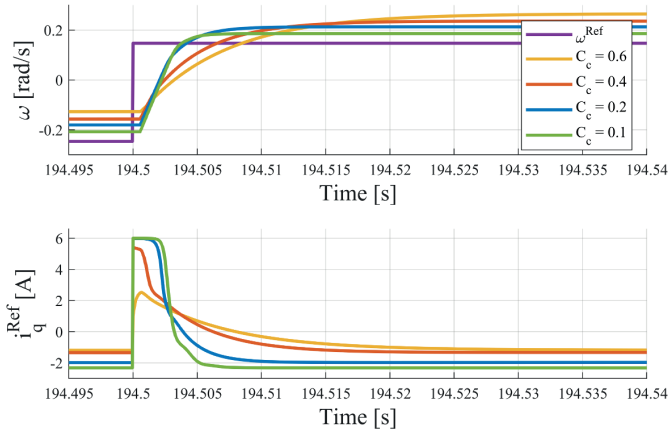


Fig. 16. Influence of C_c on the control system response

Performed research confirm the intuitive selection of parameters and stability of the proposed solution. Unfortunately, the high value of the fixed error disqualifies the network structure presented in Fig. 10. To counteract this effect, a network with integration of one of the two outputs was designed (Fig. 11). It should be noted here that in the presented case the critic becomes a two-element vector, which is calculated using the following formula:

$$f_{cr} = C_e\left(\omega^{\text{Ref}} - \omega\right) - C_c \begin{bmatrix} ANN_1^{\text{out}} \\ ANN_2^{\text{out}} \end{bmatrix} \tag{8}$$

where: ANN_m^{out} is m'th output from the network.

Figure 17 depicts the comparison between systems controlled by network without integration (Fig. 10) and with integration (Fig. 11). The network with integration correctly reduces the fixed error but introduces an overshot. In modern speed control algorithms overshot does not disqualify a controller since it can be easily fixed by e.g. applying input filters, limiting the step derivative or methods such as Input Shaping. Because of the better characteristics of the neural controller with the integration, further considerations will refer only to this topology.

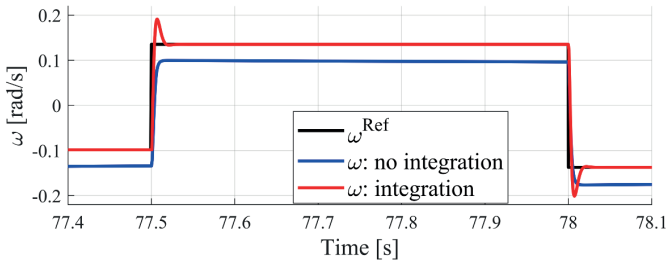


Fig. 17. Control system response for ANN speed controller with and without integration

One of the most important issues connected with the machine intelligence methods is testing the convergence of the result to the global/local minimum. In order to perform its verification, authors decided to present the characteristic of changes in the Integral Square Error quality indicator, occurring in the researched time period. The indicator can be calculated as:

$$ISE = \int_{kT_{st}}^{(k+1)T_{st}} \left(\omega^{\text{Ref}} - \omega\right)^2 \text{d}t \tag{9}$$

where: T_{st} – period of reference step, k – step number.

Teaching the network with use of pseudo-random signal and controlling an object with an interfering signal and parameters varying in time, makes it rather difficult to properly analyse obtained results. Taking the above into consideration, testing the convergence was performed on a plant with constant parameters and ω^{Ref} generated in form of steps with a constant amplitude and period (Fig. 18). The coefficient value calculated in a time period can be seen in Fig. 19. A constant decrease of the value in time is clearly visible, which proves the convergence of the algorithm.

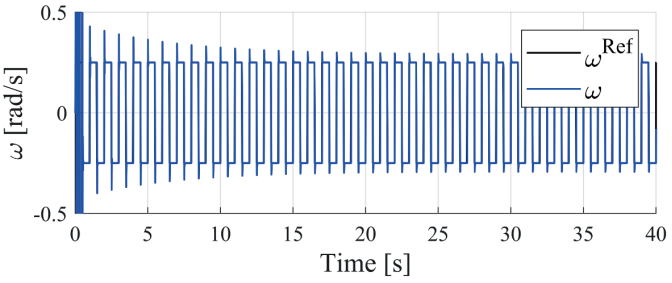


Fig. 18. Reference signal and measured speed signal for the convergence test of network learning algorithm

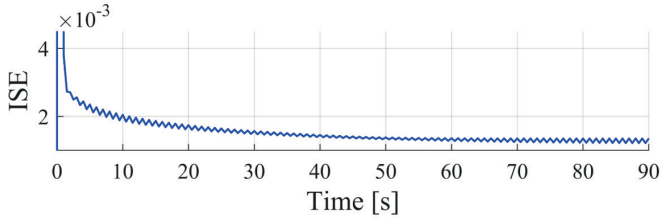


Fig. 19. ISE indicator change in time, after the drive start-up with constant parameters

Adaptive systems have the advantage of being able to retune themselves if the parameters of the object change. To prove

that, the presented system has this kind of characteristics, the previous test was repeated with the difference that in 7th and 14th second, the moment of inertia of the plant was stepwise changed. The test results can be seen in Fig. 20. As one can notice, the ISE indicator decreased each time the parameter changed in the object. The indicator setting at a higher level after increasing the J parameter is a natural feature of the system. The bigger the J the lower the motor acceleration, which increases the ISE.

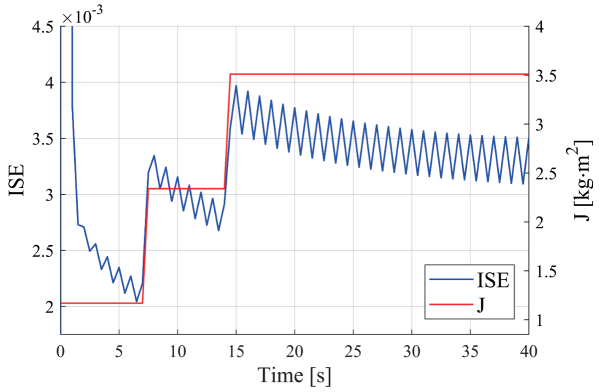


Fig. 20. ISE indicator changes in time, after the drive start-up with stepwise variable parameters

The last simulation research consisted in comparing the performance of the proposed neural controller to a PI, given by the following transmittance:

$$G_{PI}(z) = K_P\left(1 + \frac{T_s}{T_I(z-1)}\right) \tag{10}$$

where: K_P – proportional gain, T_I – integration time, T_s – sample time.

The controller used in the comparison has constant parameters that cannot change in response to variations in the moment of inertia in the object. For this reason, it was decided to tune the PI controller to the arithmetic mean of inertia. The speed controller parameters were calculated according to the symmetry criterion as follows [27]:

$$K_P = 0.5\frac{J_{\min} + J_{\max}}{2K_T T_{\mu i}}, \tag{11}$$

$$T_I = 4T_{\mu i} \tag{12}$$

where: $J_{\min/\max}$ – minimal/maximal moment of inertia.

The comparison was conducted on an object with the load torque and moment of inertia changing in the function of an angle. In the Fig. 21 one can see the step response of the PI and neural controllers. In the 78.5th second the plant had $J = 1.17$ kgm^2, and in 54th second $J = 1.98$ kgm^2. The range of the full variability of J stretched from 1.17 kgm^2 to 3.77 kgm^2. For small values of J, the PI controller operated in a way unacceptable in a real system. Vibrations created during the control would have reduced the life expectancy of mechanical components and could lead to damaging the device. In contrast to the PI controller, the neural network shows the same character of the step response regardless of the inertia.

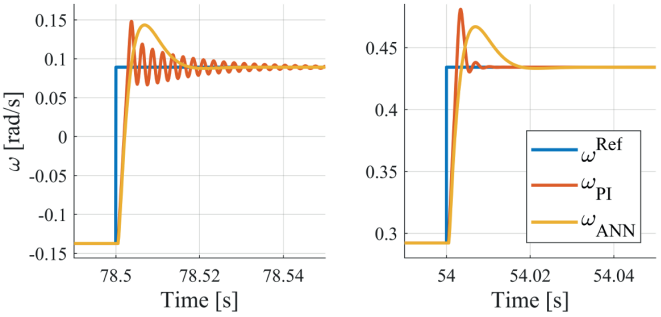


Fig. 21. Step responses of neural and PI controllers, for an object with a changeable moment of inertia and load torque

Rapid operation of the controller promotes the reduction of the ISE index by accelerating the response (Fig. 22). Due to the high similarity of the values and the illegibility of the plot, authors decided to present the difference between the indicators (Fig. 23). First of all, the correlation between the changing

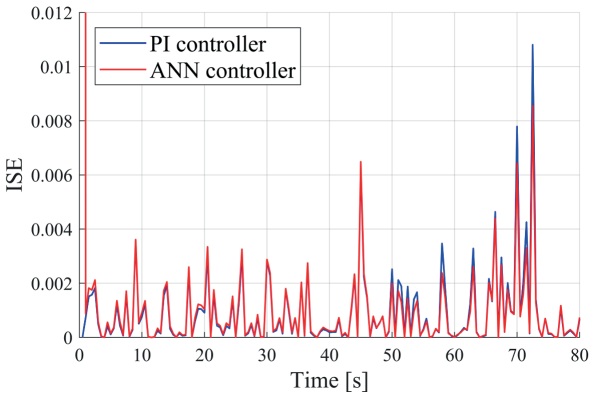


Fig. 22. ISE indicator value for both the neural and PI controllers, for an object with a changeable moment of inertia and the load torque

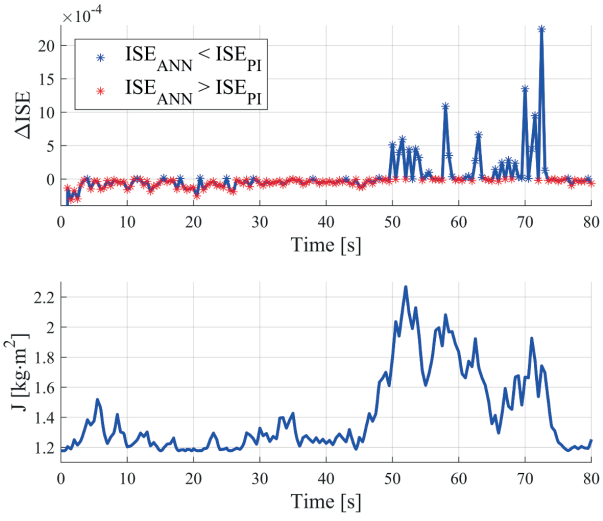


Fig. 23. Differences between the ISE indicators for the neural and PI controllers along with the moment of inertia

moment of inertia and the value of the indicator can be seen. When analysing the obtained data, one could conclude that the PI controller is, on average, better than the neural controller, however, in order to be able to truly compare the two systems, a quality indicator correlated with the critic's function should be used (6). In the research, it was decided to verify the IAEAC indicator given by the formula (13):

$$IAEAC = \int_{kT_{st}}^{(k+1)T_{st}} \left(C_e \text{abs}\left(\omega^{\text{Ref}} - \omega\right) + C_c \text{abs}\left(i_q^{\text{Ref}}\right)\right) \text{d}t. \quad (13)$$

The indicator given by (13) is presented in Fig. 24. When the cost of control is also considered during the analysis, it turns out that the value of the indicator for ANN is 6 times lower than the indicator calculated for PI.

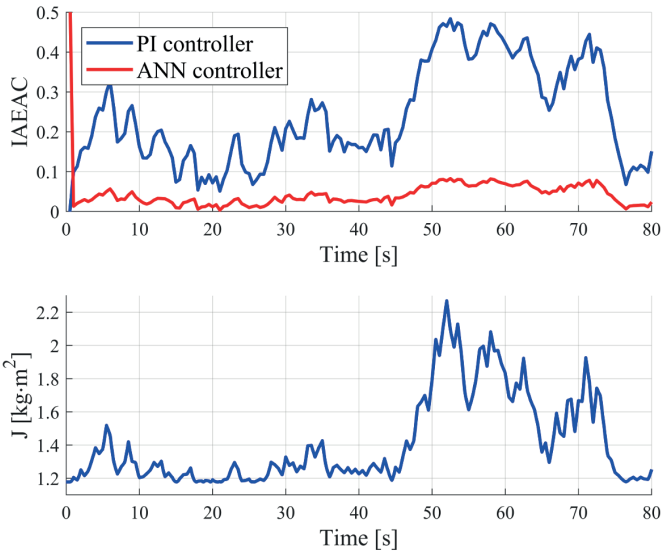


Fig. 24. IAEAC indicator value for the neural and PI controllers, for the object with a changeable moment of inertia and load torque

6. Real-plant results

The experimental tests were carried out on a laboratory driving stand equipped with a permanent magnet synchronous motor. The motor is controlled by a three-phase voltage converter from the LABINVERTER family, type P3-5.0 / 550MFE, supplied from a three-phase network with a voltage of 400 V and a frequency of 50 Hz. The control algorithm has been implemented on a ADSP-21060 signal processor from the SHARC family (Analog Devices Inc.), with feature of 40 MHz (25 ns) instruction rate and 120MFLOPs peak performance. The sampling period of the control algorithm was 100 μs. Absolute (14 bits per revolution) and incremental (with a resolution of 16384 pulses per revolution) encoders were used to measure position and angular speed, respectively. The speed measurement was performed using a technique based on a digital filter discussed in [28]. In this work, the PLL (Phase Locked Loop) method combined with the M method has been recommended. This allows to minimize the quantization noise, which behaves similarly to a low-pass filter, but with the important advantage of a shorter delay. The acquisition of measurement and control signals was carried out using a Tektronix DPO 3014 digital oscilloscope with the possibility of recording 5 megasamples and a 100 MHz band. The control software was developed and launched using the VisualDSP++ programming environment.

The implementation of the adaptive neural controller algorithm on DSP was written in C language. The controller code was based on the authors' own library, performing two main tasks: calculation of the feedforward, multilayer artificial neural network output and modification of network weights using backpropagation algorithm [29]. The developed library allowed to compile the code regardless of available support for matrix operations and was adapted to use the CMSIS library for ARM architectures and runtime libraries for SHARC architecture. Due to the assumption of using the neural controller with a sample time of 100 us and small size of the artificial neural network (less than 20 neurons per layer), the implementation was carried out with an emphasis on computation time, at the expense of greater consumption of data memory. Due to the low-level nature of the algorithm, dynamic memory allocation was avoided. The network adaptation concerns only the value of weights, not the size of the layers, so static memory allocation is a sufficient solution – all sizes of inputs, outputs, and intermediate results are known a priori. The authors' library for neural operations works with a set of scripts implemented in the MATLAB environment, which enable quick and convenient generation of C language source files containing network structure definitions of a given size and initial weights.

To carry out experimental research, a neural controller was implemented, based on the network structure shown in Fig. 11 and the adaptation procedure with the critic given by the function (6). The paper presents the results of drive start-ups in three different mechanical configurations:

(a) With a small moment of inertia, J_{min},

(b) With a large moment of inertia, J_{max},

(c) With an indirect, asymmetrical moment of inertia, J_{asym}.

Figure 25 shows waveforms of the reference and measured angular speed of the drive, as well as the reference current in q axis during the start-up of the neural controller. After just a few hundred samples, the adaptation allows to correctly track step changes in the reference value. The nature of the responses to single step changes is not yet consistent – weights adaptation is still going on.

Figure 26 shows object and controller responses waveforms after approx. 2000 seconds of constant operation of the control and adaptation algorithm. The system operates in a stable way and obtains a uniform character of the response to the step changes in the reference value, which confirms the correct performance of the controller.

Figure 27 depicts the control system response to single reference value step changes by approx. 2000 seconds from the beginning of adaptation. The neural controller ensures the ex-

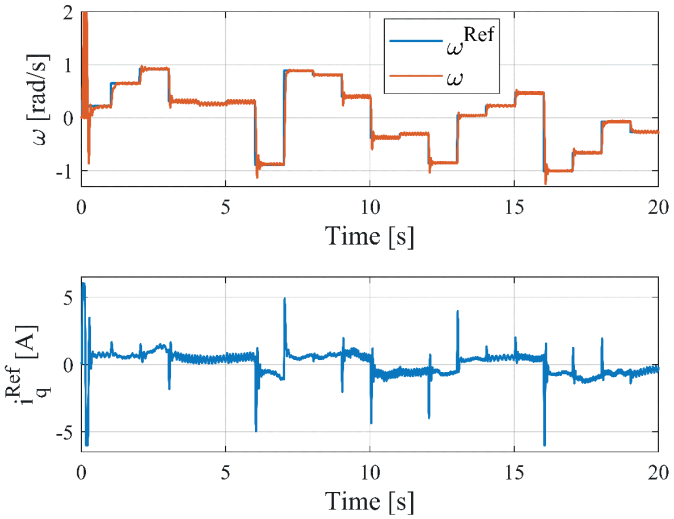


Fig. 25. Starting the system: speed and control signal waveforms at the beginning of adaptation. Operating with moment of inertia $J_{\min}$

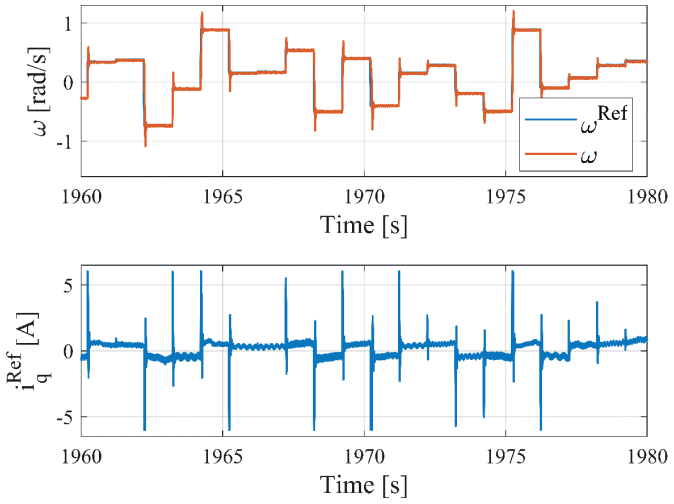


Fig. 26. Operating after approx. 2 thousand seconds: speed and control signal waveforms. Operating with moment of inertia $J_{\min}$

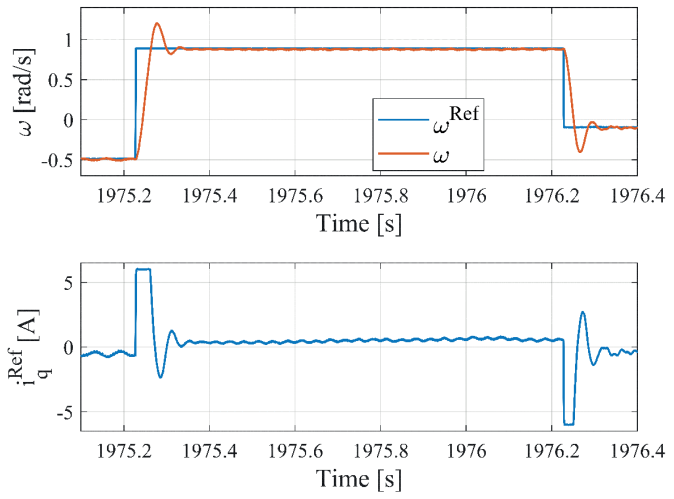


Fig. 27. System response to the reference value step change after approx. 2 thousand seconds of performance: speed and control signal waveforms. Operating with moment of inertia $J_{\min}$

pected control quality, consistent with the simulation results in the long period of adaptation.

During the experimental verification of the control algorithm, it was checked whether the system that underwent adaptations for a given configuration (set of system parameters) is able to properly adapt to the step change of parameters. To do so, the controller adaptation for a mechanical configuration with the moment of inertia $J_{\min}$ was performed. Next, the control and adaptation process was stopped, the moment of inertia was changed to $J_{\max}$, and the control and adaptation were started again. Figure 28 shows the system response after reboot.

Figure 29 shows the response of the control system to a step change in the reference value, after approx. 180 seconds of

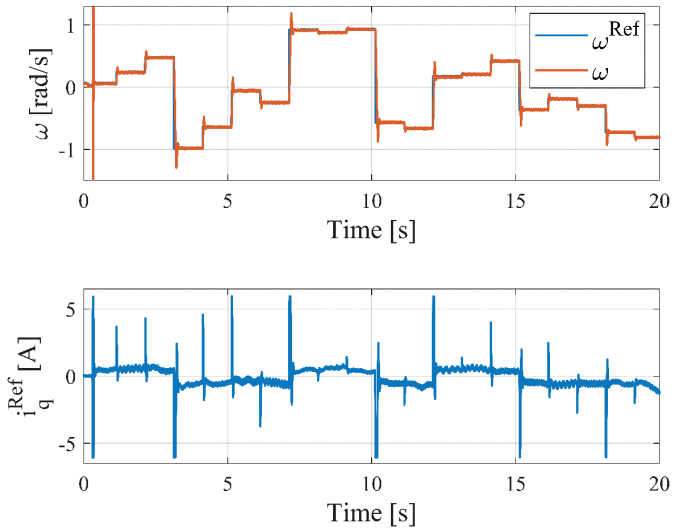


Fig. 28. Changing the moment of inertia; speed and control signal waveforms after changing the moment of inertia from $J_{\min}$ to $J_{\max}$. The system initially adapted to the configuration with $J_{\min}$

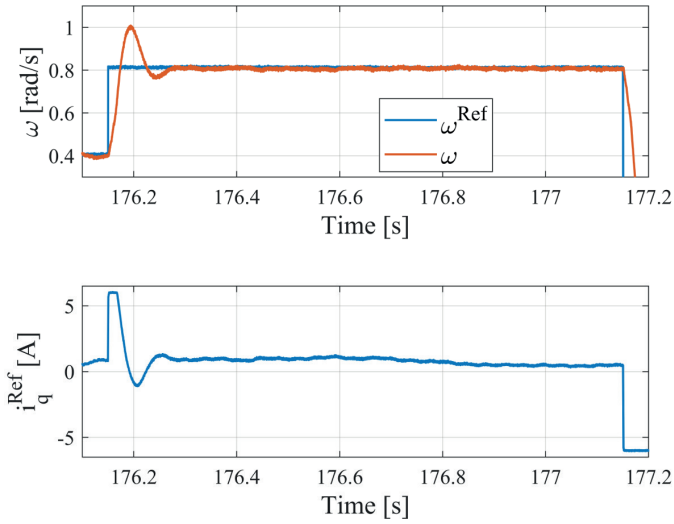


Fig. 29. Changing the moment of inertia; speed and control signal waveforms after changing the moment of inertia from $J_{\min}$ to $J_{\max}$. The system response to the reference value step change after approx. 180 seconds of the performance

operation from the moment of the system start after the change in the moment of inertia. The control system correctly adapts to changes in the drive system, obtaining a response similar to the system's response before changing the parameters.

An analogous experiment was carried out for the change of symmetry of the moment of inertia: the controller was adapted for mechanical configuration with the moment of inertia $J_{\max}$, and then the process of control and adaptation was stopped, the moment of inertia was changed to J_{asym}, and control and adaptation were started again. Asymmetrical load is characterized by a clear correlation between the load torque influencing the motor and the angular position. In an asymmetrical configuration, the drive draws a positive current when lifting the heavier side of the load (load acceleration) and a negative current when the heavier side of the load falls (load deceleration). Figure 30 shows the waveforms of speed, control signal and motor shaft positions for an unbalanced load. Based on the system's response, it can be concluded that the controller has correctly adapted to the changes in system parameters. The relationship between the control signal and the position of the motor shaft is also clearly visible.

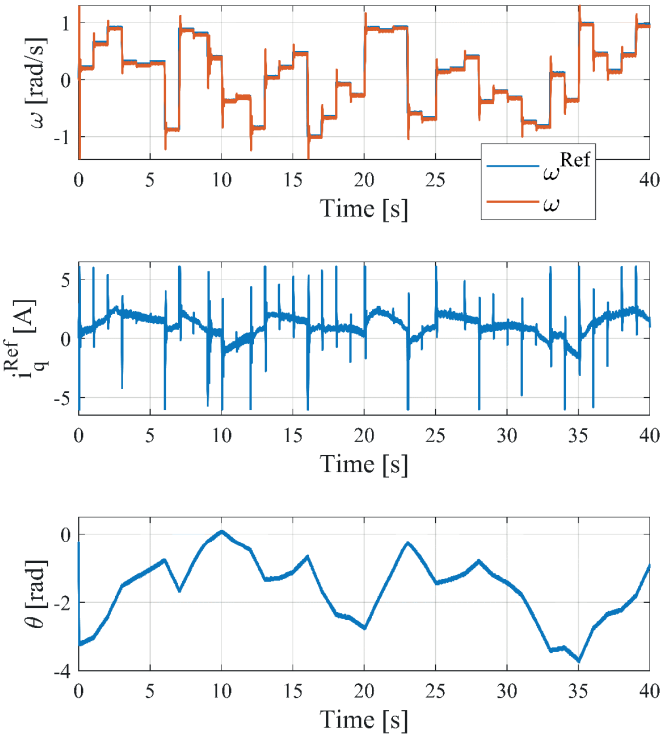


Fig. 30. Changing the symmetry of the moment of inertia: speed and control signal waveforms after changing the moment of inertia from $J_{\max}$ to J_{asym}. The system initially adapted for the configuration with $J_{\max}$

Figure 31 presents the results of neural controller adaptation during the system start, for a mechanical configuration with an asymmetric moment of inertia J_{asym}. Also, in this case the system adapts the weights of the network in a correct way and obtains an analogous dependency between the angular position and the control signal.

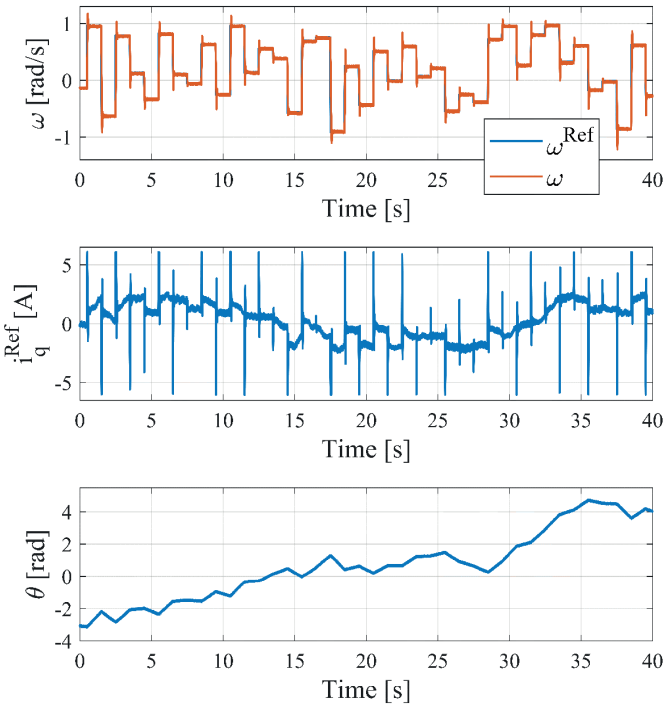


Fig. 31. Starting the system: speed and control signal waveforms at the beginning of adaptation. Performance with an asymmetric moment of inertia J_{asym}

7. Conclusion

The paper presents a method of designing a neural speed controller with the use of Reinforcement Learning method. The controlled object was an electric drive with a synchronous motor with permanent magnets, of a complex mechanical structure and changeable parameters. The proposed method, by the principle, does not require any information about the environment to be able to change policies of the agent to realize given tasks. The simulation and real tests show that proposed system can control an object with variable inertia and load torque, which correlate with the angle of the shaft (rotor). The presented critic allows for a stable operation of the system even after a very long adaptation time, which was also verified on a real stand. The controller maintains the character of its response independently to the changes in the object's moment of inertia. Moreover, it is more cost-saving than the PI controller. The obtained research results encourage further work, which should focus on various forms of the critic's functions and the stability of solutions in long time horizons.

Acknowledgements. This work has been founded by Poznan University of Technology under project 04/45/SBAD/0210.

References

[1] A. Babiarz, J. Klamka, R. Bieda, and K. Jaskot, "The dynamics of the human arm with an observer for the capture of body motion parameters", *Bull. Pol. Ac.: Tech.* 61 (4), 955–971 (2013).

[2] J. Ikaheimo, "Permanent magnet motors eliminate gearboxes", *ABB Review* 4, 22–25 (2002).

[3] K. Wróbel, K. Szabat, and P. Serkies, "Long-horizon model predictive control of induction motor drive", *Arch. Electr.Eng.* 68 (3), 579–593 (2019).

[4] T.C. Chen and T.T. Sheu, "Model reference robust speed control for induction-motor drive with time delay based on neural network", *IEEE Trans. Syst., Man, Cybern. Syst.* 31 (6), 746–753 (2001).

[5] J. Kabziński, "Adaptive, compensating control of wheel slip in railway vehicles", *Bull. Pol. Ac.: Tech.* 63 (4), 955–963 (2015).

[6] B.K. Bose, "Neural Network applications in power electronics and motor drives—An introduction and perspective", *IEEE Trans. Ind. Electron.* 54 (1), 14–33 (2007).

[7] M. Kaminski and T. Orlowska-Kowalska, "FPGA implementation of ADALINE-based speed controller for two-mass system", *IEEE Trans. Ind. Informat.* 9 (3), 1301–1311 (2013).

[8] B. Ufnalski and L.M. Grzesiak, "Repetitive neurocontroller with disturbance feedforward path active in the pass-to-pass direction for a VSI inverter with an output LC filter", *Bull. Pol. Ac.: Tech.* 64 (1), 115–125 (2016).

[9] L.M. Grzesiak, V .Meganck, J. Sobolewski, and B. Ufnalski, "On-line trained neural speed controller with variable weight update period for direct-torque-controller AC drive", *12th International Power Electronics and Motion Control Conference*, Portoroz, 1127–1132 (2006).

[10] T. Orlowska-Kowalska and K. Szabat, "Control of the drive system with stiff and elastic coupling using adaptive neuro-fuzzy approach", *IEEE Trans. Ind. Electron.* 51 (4), 228–240 (2007).

[11] M.A. Rahman and M.A. Hoque, "On-line adaptive artificial neural network based vector control of permanent magnet synchronous motors", *IEEE Trans. Energy Convers.* 13 (4), 311–318 (1988).

[12] D. Chen and M. York, "Adaptive neural inverse control applied to power systems", *IEEE PES Power Systems Conference and Exposition*, Atlanta, GA, 2109–2115 (2006).

[13] E. Colina-Morles and N. Mort, "Inverse model neural network-based control of dynamic systems", *International Conference on Control – Control '94*, Coventry, UK, 955–960 vol. 2 (1994).

[14] Y. Li, B. Zhang, and X. Xu, "Decoupling control for permanent magnet in-wheel motor using internal model control based on back-propagation neural network inverse system", *Bull. Pol. Ac.: Tech.* 66 (6), 961–972 (2018).

[15] T.M. Mitchell, *Machine Learning*, McGraw-Hill Science/Engineering/Math, 1997.

[16] L. Dong Jin and B. Hyochoong, "Model-free LQ control for unmanned helicopters using reinforcement learning", *11th International Conference on Control, Automation and Systems*, Gyeonggi-do, 117–120 (2011).

[17] D. Lee, M. Choi, and H. Bang, "Model-free linear quadratic tracking control for unmanned helicopters using reinforcement learning", *The 5th International Conference on Automation, Robotics and Applications*, Wellington, 19–22 (2011).

[18] Xiao-ting Cui and Xiang-dong Liu, "Fuzzy Neural Control of Satellite Attitude by TD Based Reinforcement Learning", *6th World Congress on Intelligent Control and Automation*, Dalian, 3983–3986 (2006).

[19] J. Xue, Q. Gao, and W. Ju, "Reinforcement Learning for Engine Idle Speed Control", *International Conference on Measuring Technology and Mechatronics Automation*, Changsha City, 1008–1011 (2010).

[20] E. Bejar and A. Moran, "Deep reinforcement learning based neuro-control for a two-dimensional magnetic positioning system", *4th International Conference on Control, Automation and Robotics (ICCAR)*, Auckland, 268–273 (2018).

[21] B. Subudhi and S.K. Pradhan, "Direct adaptive control of a flexible robot using reinforcement learning", *International Conference on Industrial Electronics, Control and Robotics*, Orissa, 129–136 (2010).

[22] H. Li, "The implementation of reinforcement learning algorithms on the elevator control system", *IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, Luxembourg, 1–4 (2015).

[23] S. Zhipeng, G. Chen, and S. Jianbo, "Reinforcement learning control for ship steering based on general fuzzified CMAC", *5th Asian Control Conference (IEEE Cat. No.04EX904)*, Melbourne, Victoria, Australia, 1552–1557 vol. 3 (2004).

[24] S. Bhasin, *Reinforcement learning and optimal control methods for uncertain nonlinear systems*, Dissertation Ph.D, University of Florida, (2011).

[25] A. Barto, A. G., Sutton, and C. Anderson, "Neuron-like adaptive elementsthat can solve difficult learning control problems", *IEEE Trans. Syst., Man, Cybern. Syst.* 13 (5), 834–846 (1983).

[26] T. Pajchrowski, K. Zawirski, and K. Nowopolski, "A Neural Speed Controller Trained On-Line by Means of Modified RPROP Algorithm", *IEEE Trans. Ind. Inform*. 11 (2), 560–568 (2015).

[27] J. W. Umland and M. Safiuddin, "Magnitude and symmetric optimum criterion for the design of linear control systems: what is it and how does it compare with the others?", *IEEE Trans. Ind Appl.* 26 (3), 489–497 (1990).

[28] D. Łuczak, K. Nowopolski, K. Siembab, and B. Wicher, "Speed calculation methods in electrical drive with non-ideal position sensor", *19th International Conference on Methods and Models in Automation and Robotics (MMAR)*, Międzyzdroje, 726-731 (2014).

[29] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks", *Chemom. Intell. Lab. Syst.* 39 (1), 43–62 (1997).