Jakub MOŻARYN, Jerzy E. KUREK
POLITECHNIKA WARSZAWSKA , INSTYTUT AUTOMATYKI I ROBOTYKI

Using Tikhonov regularization to improve estimation of robot position based on uncertain robot model obtained by neural network

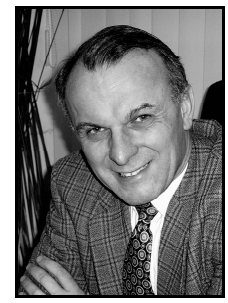
Mgr inż. Jakub MOŻARYN

Jest asystentem w Instytucie Automatyki i Robotyki Politechniki Warszawskiej. Ukończył studia w roku 2001 na Wydziale Mechatroniki Politechniki Warszawskiej na specjalności robotyka. Do jego głównych zainteresowań naukowych należą m.in. sztuczna inteligencja, robotyka, układy sterowania.

e-mail: j.mozaryn@mchtr.pw.edu.pl

Prof. dr hab. inż. Jerzy KUREK

Profesor zwyczajny Pol. Warszawskiej (PW). W kadencji 2002-2005 Dziekan Wydziału Mechatroniki PW. Absolwent Wydziału Mechaniki Precyzyjnej PW (obecnie Wydział Mechatroniki). Jego zainteresowania naukowe dotyczą m. in. wyznaczania nowych algorytmów sterowania, stabilności układów automatyki, wyznaczania modeli neuronowych układów dynamicznych, szczególnie robotów. Jest członkiem Control System Society IEEE, Inc. oraz członkiem zarządu Polskiego Stowarzyszenia Pomiarów, Automatyki i Robotyki POLSPAR.



e-mail: jkurek@mchtr.pw.edu.pl

Abstract

A method for improvement of a position estimation of a robot manipulator based on model with uncertain parameters is presented. To calculate the position of the robot there was designed the robot model using artificial neural networks with structure of the mathematical model in the form of *Lagrange-Euler* equations. The *Tikhonov* regularization was then used to improve the approximation of the robot's position. The example of the position of the robot PUMA 560 with 6 degrees of freedom calculation with proposed method is presented. Obtained results indicate significant improvement of the estimation.

Keywords: robotics, neural networks, inverse dynamic problem, regularization.

Zastosowanie regularyzacji Tikhonova do poprawy estymacji pozycji robota na podstawie modelu o niedokładnych parametrach wyznaczonych za pomocą sieci neuronowych

Streszczenie

W pracy przedstawiono metodę poprawy estymacji położeń robota na podstawie modelu robota o niedokładnych parametrach. Do wyznaczania położenia robota zaprojektowano model robota z wykorzystaniem sztucznych sieci neuronowych o strukturze modelu matematycznego w formie równań *Lagrange'a-Eulera*. W celu poprawy estymacji położeń na podstawie wyznaczonego modelu zastosowano regularyzację *Tikhonowa*. Zaproponowana metoda została przedstawiona na przykładzie odtwarzania położeń robota PUMA 560. Otrzymane wyniki wskazują na znaczną poprawę dokładności.

Słowa kluczowe: robotyka, sieci neuronowe, model robota, regularyzacja.

1. Introduction

Mathematical model of robot has complicated, highly nonlinear multi-input multi-output structure. Equations describing the dynamics of the robot can be calculated using the *Lagrange-Euler* equation [2], but this requires a knowledge of the exact values of the robot's physical parameters that are hard to obtain [4].

It is possible to identify the robot model with neural networks which parameters can be easily, adaptively calculated. Different neural networks presented in [6, 10, 12, 13] are examples of such model. Designing neural networks does not require the knowledge of functions of the model, but only values of robot joints position. Moreover the structure of the neural network can resemble the model of the robot and identify elements in the *Lagrange-Euler* equation. However, there exists a problem of an uncertainty of identified parameters which results in sudden decrease of the position estimation accuracy [12]. This can be caused for instance by loss of the positive definiteness of the identified inertia matrix, it was investigated in [13].

One of the techniques for improvement of approximation based on uncertain model is regularization. The most common is the *Tikhonov* regularization [1, 3, 7, 8] also known as the *Tikhonov-Powell* regularization or the ridge regression. In this paper using the *Tikhonov* regularization we present a method for improvement of the robot manipulator position estimation calculated based on uncertain neural robot model.

The article is organized as follows. In section 2 a discrete time mathematical model of robot manipulator in the form of the *Lagrange-Euler* equation is described, and the structure of neural network for identification of the inverse robot model is presented. Afterwards, the robot position estimation using identified neural model is described. The *Tikhonov* regularization and properties of ill-posed problems are presented in section 3 and in section 4 the robot position estimation based on neural network model with the *Tikhonov* regularization is presented. Section 5 describes numerical results obtained for estimation of the PUMA 560 robot manipulator position. Finally, concluding remarks are given.

2. Neural network Lagrange-Euler robot model

The discrete time model of the robot with n degree of freedom, based on the *Lagrange-Euler* equation [2] can be presented as follows [12]

$$\tau(k) = V(k) + G(k) + M(k)T_p^{-2}[q(k+1) - 2q(k) + q(k-1)] \quad (1)$$

where $\tau(k) = [\tau_i(k)] \in R^n$ is vector of control signals, $q(k) = [q_i(k)] \in R^n$ is vector of generalized joint coordinates, $M(k) = [m_{ij}(k)] \in R^{n \times n}$ is robot inertia matrix, $M(k) = M[q(k)]$, $V(k) \in R^n$ is vector of *Coriolis* and centrifugal forces, $V(k) = V[q(k), q(k-1)]$, $G(k) \in R^n$ is vector of the gravity loading, $G(k) = G[q(k)]$, k is discrete time and T_p is sampling period ($t = kT_p$).

The above model can be rewritten as follows [12]

$$\tau(k) = P(k) + M(k)T_p^{-2}[q(k+1) - 2q(k) + q(k-1)] \quad (2)$$

where

$$P(k) = [p_i(k)] = V(k) + G(k) \quad (3)$$

In the robot model (1) the unknown nonlinear elements of $M(k)$, $V(k)$, $G(k)$ should be identified. For identification we present model (2) as a set of n equations

$$\tau_i(k) = p_i(k) + \sum_{j=1}^{n} m_{ij}(k) T_p^{-2} [q_j(k+1) - 2q_j(k) + q_j(k-1)] \quad (4)$$

$$i = 1, \ldots, n$$

Elements $p_i(k)$ and $m_{ij}(k)$ can be identified using a feed-forward neural network [12]. Structure of the neural network for the identification of elements is shown in Fig. 1(a). Inputs to the neural network are generalized joint coordinates $q(k)$ and the output of the network is $\tau_i(k)$. Therefore we call the obtained model the 'inverse' one. Clearly, we have n independent neural models (4) for every degree of freedom.

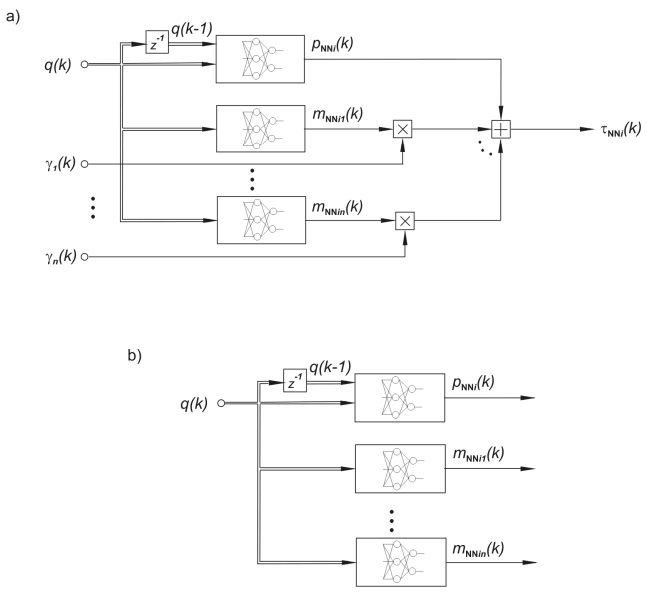


Fig. 1. The structure of neural network for (a) identification and (b) estimation of elements in model (4)
Rys. 1. Struktura sieci neuronowej (a) do identyfikacji i (b) do estymacji elementów w modelu (4)

The identified model can be used for estimation of unknown elements of the robot model based only on joint coordinates $q(k)$, see Fig. 1(b). It is easy to note that matrices in (2) can be estimated as follows

$$M(k) \cong M_{NN}(k) = [m_{NNij}(k)], \quad i = 1, \ldots, n, \; j = 1, \ldots, n \quad (5)$$

$$P(k) \cong P_{NN}(k) = [p_{NNi}(k)], \quad i = 1, \ldots, n \quad (6)$$

where elements $m_{NNij}(k)$ and $p_{NNi}(k)$ are calculated by neural network according to Fig. 1(b).

Then, the following equation for estimation of robot position can be used

$$q_{NN}(k+1) = 2q(k) - q(k-1) + \Lambda_{NN}(k) \quad (7)$$

where $q_{NN}(k+1)$ is estimated position and

$$\Lambda_{NN}(k) = T_p^2 M_{NN}^{-1}(k)[\tau(k) - P_{NN}(k)] \quad (8)$$

3. The Tikhonov regularization

In many automatic control applications occurs a problem of finding a good approximation of the vector $x \in R^n$ that satisfies a standard matrix linear equation

$$Ax = b \quad (9)$$

where $A = [a_{ij}] \in R^{n \times n}$ is matrix of approximated coefficients, $x = [x_i] \in R^n$ represents vector of unknown parameters and $b = [b_j] \in R^n$ is vector of measurements.

The presented problem is called well-posed in the sense of *Hadamard* if following conditions are satisfied [3, 7, 8]: a) a solution exists, b) the solution is unique, c) the solution depends continuously on the data. If one of presented conditions is not satisfied the problem is called ill-posed.

Usually, if (9) is ill-posed, the solution has very big errors. In such case matrix A is also ill-conditioned. This property can be tested using a condition number which can be calculated according to a following equation [5]

$$\kappa_A = \frac{\sigma_{A\max}}{\sigma_{A\min}} \quad (10)$$

where $\sigma_{A\max}, \sigma_{A\min}$ are maximal and minimal singular values of matrix A [5].

Matrices with the condition number near 1 are said to be well-conditioned and matrices with the condition number much above 1 are said to be ill-conditioned.

If the problem is ill-posed, during the numerical solution it can be regularized using additional assumptions. The commonly used method is the *Tikhonov* regulatization [3, 7, 8]. Regularized solution is defined as follows

$$\hat{x} = \arg\min_x \{\|Ax - b\|_2^2 + \alpha^2 \|\Gamma x\|_2^2\} \quad (11)$$

where $\hat{x}$ is explicit solution , $\Gamma \in R^{nxn}$ denotes *Tikhonov* matrix (it is assumed that $\Gamma^T \Gamma$ is positive definite), $\alpha > 0$ is scaling factor, $\|x\|_2$ denotes *Euclidean* norm of vector x.

Solving (11) leads to the following formula, derived by *Tikhonov* [1]

$$\hat{x} = (A^T A + \alpha \Gamma^T \Gamma)^{-1} A^T b \quad (12)$$

Usually, in the simple approximation, Γ can be chosen as an identity matrix [7, 8]. Scaling factor α can be chosen arbitrarily or by using special techniques, for instance cross validation, maximum likelihood, conjugate gradients or evolutionary algorithms [7, 8, 11].

4. The position estimation of the robot using neural network model and the Tikhonov regularization

In every step k the identified by neural network model of robot can be rewritten using (2) and identified matrices (5)-(6) into the matrix form of linear equations as follows

$$M_{NN}(k) T_p^{-2} [q_{NN}(k+1) - 2q(k) + q(k-1)] = b_{NN}(k) \quad (13)$$

where

$$b_{NN}(k) = \tau(k) - P_{NN}(k) \quad (14)$$

Using the *Tikhonov* regularization according to (12) with the Tikhonov matrix $\Gamma = I$, the following equation for the position estimation can be used

$$q_{NN}(k+1) = 2q(k) - q(k-1) + \Lambda_{NNr}(k) \tag{15}$$

where

$$\Lambda_{NNr}(k) = T_p^2 [M_{NN}^T(k) M_{NN}(k) + \alpha I]^{-1} M_{NN}^T(k)[\tau(k) - P_{NN}(k)] \tag{16}$$

5. Numerical example

It was shown in [12] that calculation of robot position estimate according to (7) can give big errors, e.g. for robot PUMA 560 [3, 7].

In the example the PUMA 560 robot was simulated in the time interval T = 200 [sec], with the sampling time T_p = 0.01 [sec]. Therefore, there were 20 000 data samples for the training and 20 000 data samples for the testing of neural networks. The training and testing data, reference trajectories and control inputs, were set different for every joint. Both trajectories are presented in Fig. 2-3 and it is easy to see the differences.

Every element of the robot's mathematical model (4) was identified by a two layer neural network with one nonlinear hidden layer with 2 neurons and one linear output layer with 1 neuron. In all nonlinear layers neurons were described by sigmoidal activation function

$$y = f_{NL}(u) = \text{tansig}(u) \tag{17}$$

In linear layers neurons were described with a linear activation function

$$y = f_L(u) = u \tag{18}$$

where

$$u = w_0 + \sum_{i=1}^{N} w_i u_i \,, \tag{19}$$

and N is a number of a neuron inputs, w_i denotes a weight of the i-th input to the neuron, u_i is an i-th input to the neuron and w_0 is a threshold offset.

Neural networks were trained off-line using the conjugate gradient method to update weights in all layers [14]. There were 100 training iterations. The neural network training was carried out based on the training trajectory. The performance function of the neural network was chosen as a mean square error of the control signal in i-th joint

$$J_i = \frac{1}{K} \sum_{k=1}^{K} [\tau_i(k) - \tau_{NNi}(k)]^2 \tag{20}$$

where K is a number of all data samples.

Then, network generalization properties were tested using the testing trajectories. Both trajectories were different for every joint, compare Fig. 2 and 3. The accuracy of the position estimation in every joint for learning and testing trajectories was evaluated using the following quality indices

a) average absolute position error

$$e_{avi} = \frac{1}{K} \sum_{k=1}^{K} \left| q_{NNi}(k) - q_{ri}(k) \right| \tag{21}$$

b) maximum absolute position error

$$e_{\max i} = \max_{k \in [1,K]} \left| q_{NNi}(k) - q_{ri}(k) \right| \tag{22}$$

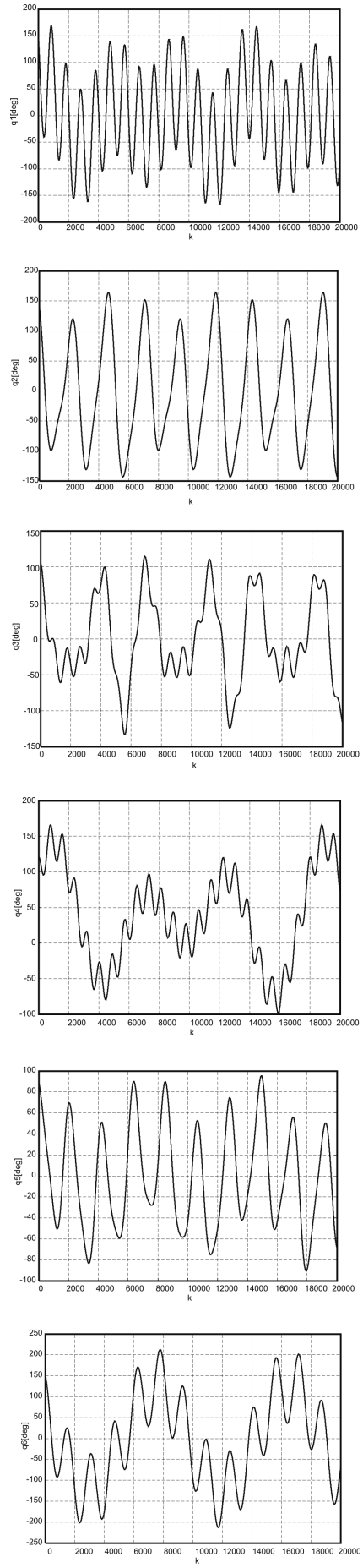
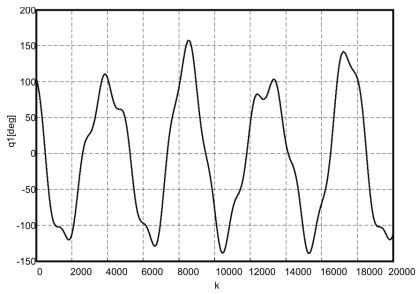
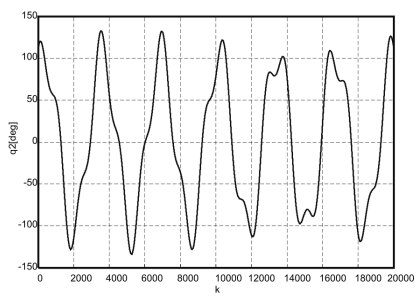
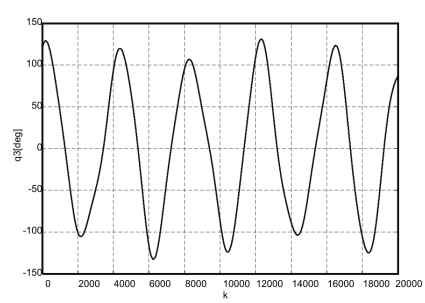
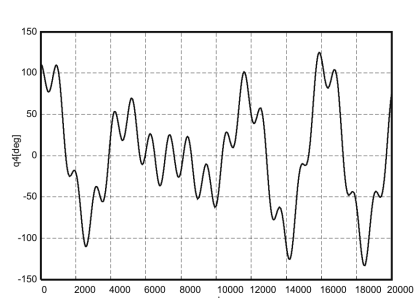


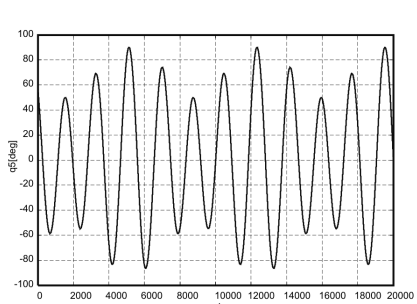
Fig. 2. Training trajectories for each joint of the robot PUMA 560
Rys. 2. Trajektorie treningowe dla poszczególnych przegubów robota PUMA 560











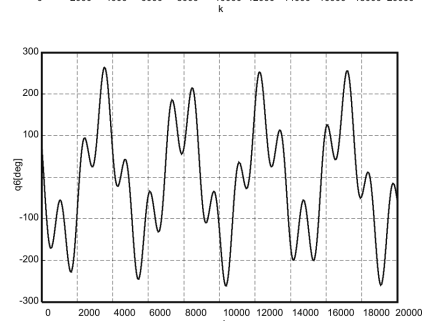
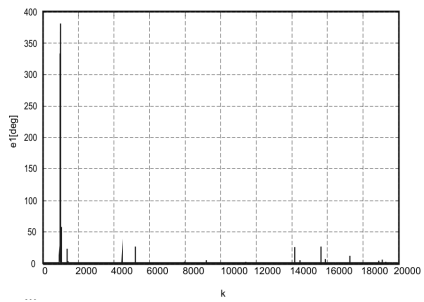
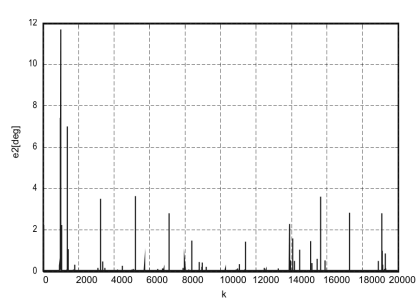
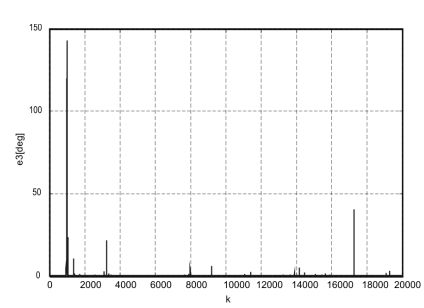
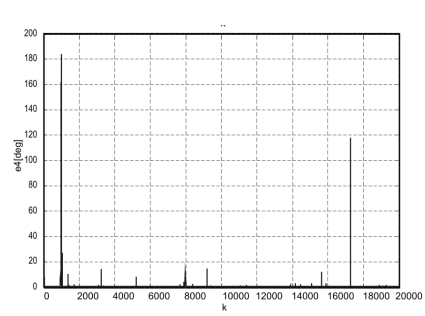


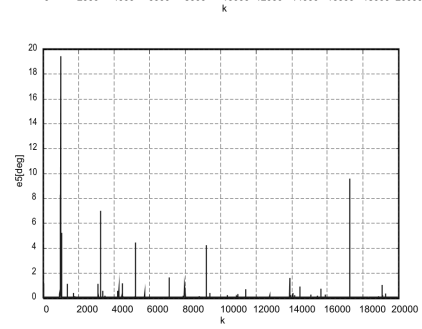
Fig. 3. Testing trajectories for each joint of the robot PUMA 560
Rys. 3. Trajektorie testowe dla poszczególnych przegubów robota PUMA 560











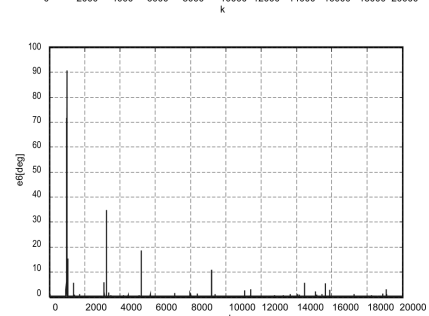


Fig. 4. Errors of robot PUMA 560 position estimation without regularization for training trajectories
Rys. 4. Błędy estymaty pozycji robota PUMA 560 dla trajektorii treningowych bez regularyzacji

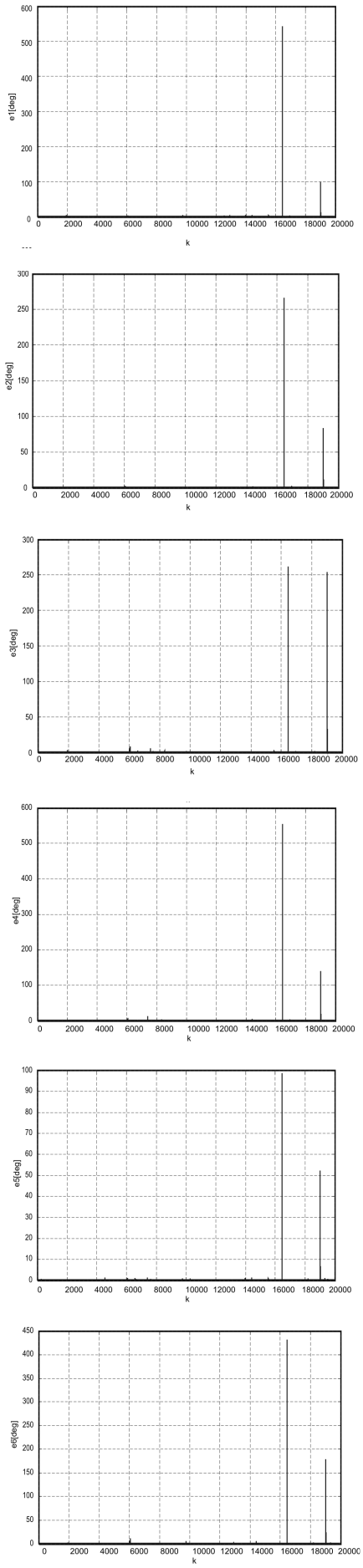


Fig. 5. Errors of robot PUMA 560 position estimation without regularization for testing trajectories

Rys. 5. Błędy estymaty pozycji robota PUMA 560 dla trajektorii testowych bez regularyzacji

Quality indices (21) - (22) calculated for each joint of the robot are given in Table 1. Errors in each joint for training and testing trajectories are presented in Fig. 4-5.

Tab. 1. Average and maximum errors for training and testing trajectories without the regularization of the estimated inertia matrix of PUMA 560 robot

Tab. 1. Średnie i maksymalne błędy dla trajektorii treningowych i testowych bez regularyzacji estymowanej macierzy inercji robota PUMA 560

Error	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Training trajectory						
e_{avi} [deg]	0.085	0.008	0.038	0.056	0.009	0.030
$e_{\max i}$ [deg]	381	11	143	184	19	91
Testing trajectory						
e_{avi} [deg]	0.052	0.026	0.042	0.055	0.013	0.048
$e_{\max i}$ [deg]	544	266	261	554	98	431

Results presented in Table 1 and in Fig. 4-5 show that the PUMA 560 position estimation was very inaccurate because maximum errors (22) were very high.

Then, we have used the proposed method with Tikhonov regularization in order to improve accuracy of the robot position estimation.

First, in every step there was calculated the condition number (10) of the matrix $M_{NN}(k)$. For the model without the regularization obtained values are presented in Fig. 6. It is easy to see that in many cases the condition number was very big and the identified matrix was ill-conditioned.

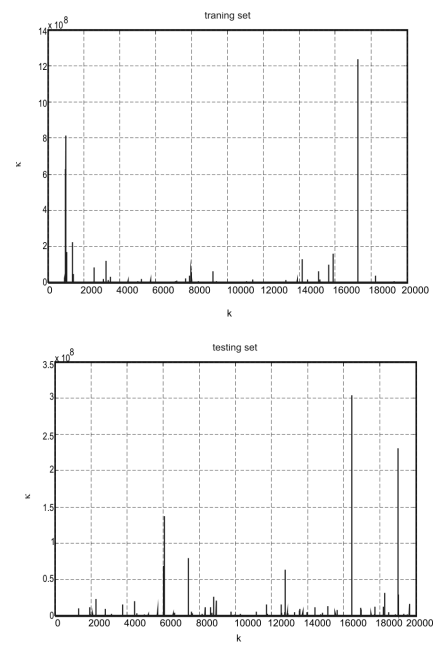


Fig. 6. The condition number of the estimated matrix for the training set and the testing set

Rys. 6. Liczba warunkowa macierzy estymowanej dla danych testowych i treningowych

Since the estimated inertia matrix was frequently ill-conditioned the presented estimation method was used. Therefore, robot positions estimates based on the obtained neural model were calculated with the regularization according to (15). Quality indices (21)-(22) obtained after the regularization with different values of the scaling factor α are given in Tables 2-4, values of the scaling factor were chosen arbitrarily equal to 1, 10 and 100.

Tab. 2. Average and maximum errors for training and testing trajectories with the regularization of the estimated inertia matrix and scaling factor α=1 for PUMA 560 robot

Tab. 2. Średnie i maksymalne błędy dla trajektorii treningowych i testowych z regularyzacją estymowanej macierzy inercji i współczynnikiem skalowania α =1 robota PUMA 560

Error	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Training trajectory						
e_{avi} [deg]	0.0021	0.0011	0.0012	0.0013	0.0011	0.0013
$e_{\max i}$ [deg]	0.0177	0.0218	0.0109	0.0134	0.0159	0.0110
Testing trajectory						
e_{avi} [deg]	0.0010	0.0012	0.0013	0.0014	0.0010	0.0019
$e_{\max i}$ [deg]	0.0094	0.0105	0.0166	0.0152	0.0104	0.0156

Tab. 3. Average and maximum errors for training and testing trajectories with the regularization of the estimated inertia matrix and scaling factor α=10 for PUMA 560 robot

Tab. 3. Średnie i maksymalne błędy dla trajektorii treningowych i testowych z regularyzacją estymowanej macierzy inercji i współczynnikiem skalowania α =10 robota PUMA 560

Error	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Training trajectory						
e_{avi} [deg]	0.0021	0.0010	0.0012	0.0012	0.0011	0.0013
$e_{\max i}$ [deg]	0.0112	0.0195	0.0084	0.0123	0.0085	0.0106
Testing trajectory						
e_{avi} [deg]	0.0009	0.0011	0.0013	0.0014	0.0010	0.0019
$e_{\max i}$ [deg]	0.0076	0.0087	0.0106	0.0111	0.0083	0.0149

Tab. 4. Average and maximum errors for training and testing trajectories with the regularization of the estimated inertia matrix and scaling factor α=100 for PUMA 560 robot

Tab. 4. Średnie i maksymalne błędy dla trajektorii treningowych i testowych z regularyzacją estymowanej macierzy inercji i współczynnikiem skalowania α =100 robota PUMA 560

Error	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
Training trajectory						
e_{avi} [deg]	0.0023	0.0008	0.0010	0.0010	0.0009	0.0011
$e_{\max i}$ [deg]	0.0061	0.0096	0.0067	0.0093	0.0058	0.0081
Testing trajectory						
e_{avi} [deg]	0.0007	0.0009	0.0010	0.0011	0.0009	0.0017
$e_{\max i}$ [deg]	0.0040	0.0057	0.0056	0.0072	0.0047	0.0102

Results presented in Tables 1 – 4 show that the accuracy of the position estimation increased significantly after the regularization. For training trajectories, depending on the joint number and the scaling factor, average errors decreased from ~7 to ~55 times and maximum errors decreased from ~500 to ~62 500 times. For testing trajectories, depending on the joint number and the scaling factor, average errors decreased from ~13 to ~55 times and maximum errors decreased from ~9 500 to ~136 500 times. One can see (Tables 2 – 4) that for the position estimation with the regularization average errors increase and maximum errors decrease if the scaling factor increase.

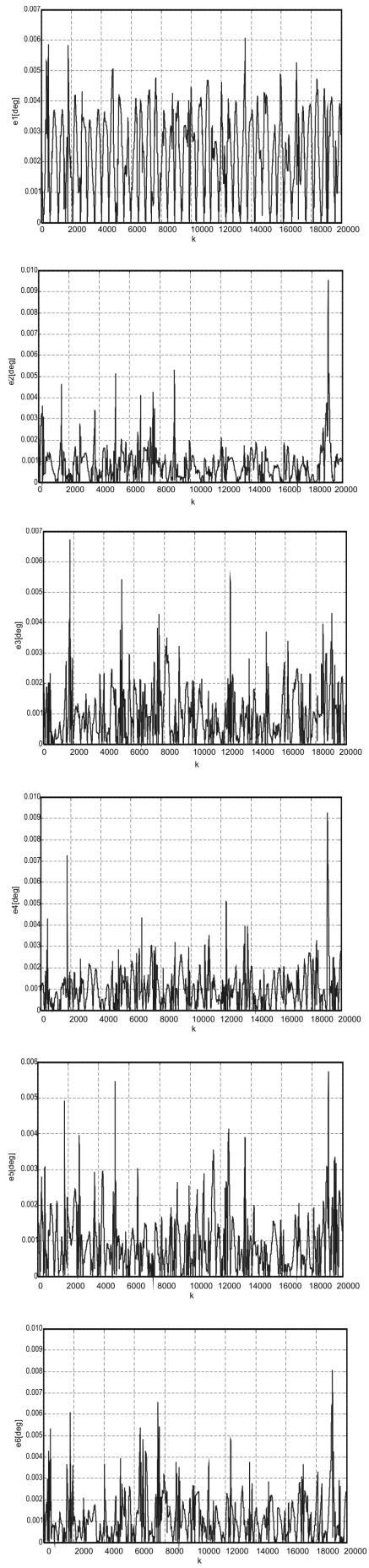


Fig. 7. Errors of robot PUMA 560 position estimation with regularization for training trajectories with scaling factor α=100

Rys. 7. Błędy estymaty pozycji robota PUMA 560 dla trajektorii treningowych z regularyzacją i ze współczynnikiem skalowania α=100

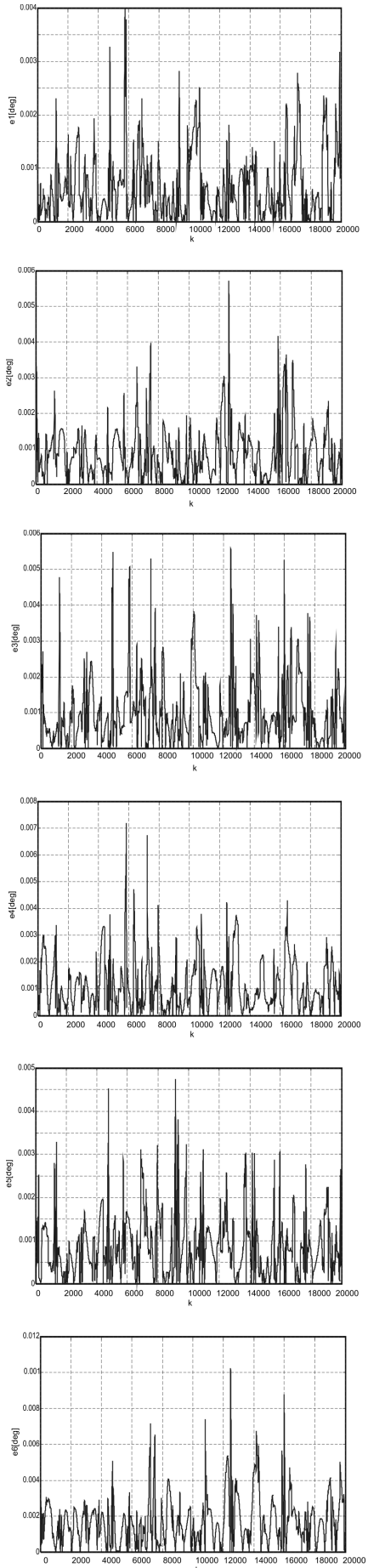


Fig. 8. Errors of robot PUMA 560 position estimation with regularization for testing trajectories with scaling factor α=100

Rys. 8. Błędy estymaty pozycji robota PUMA 560 dla trajektorii testowych z regularyzacją i ze współczynnikiem skalowania α=100

In Fig. 7-8 there are presented errors in each joint along training and testing trajectories where scaling factor $\alpha = 100$.

6. Concluding remarks

The method of the position estimation of robot manipulator based on the inverse neural model improvement was presented. The proposed technique applies the *Tikhonov* regularization for calculation of the inverse of estimated inertia matrix. Obtained results of the 6 degree of freedom robot PUMA 560 show that the proposed method significantly improves the accuracy the position estimation with presented neural networks.

However, there are certain problems that should be solved in the future. For instance further research should be done to find a proper methodology to choose the scaling factor which in presented simulations was chosen arbitrarily.

7. References

[1] A. N. Tikhonov: Solution of incorrectly formulated problems and regularization method. Soviet. Math. Dokl. 4, pp. 1035-1038, 1963.

[2] K. S. Fu, R. C. Gonzalez Lee: Robotics: control, sensing, vision and intelligence. McGraw-Hill Book Company, 1987.

[3] V. B. Glasko: Inverse Problems of Mathematical Physics. American Institute of Physics, 1988.

[4] P. I. Corke, B. Armstrong-Hélouvry: A search for consensus among model parameters reported for the PUMA 560 robot. Proc. IEEE Int. Conf. Robotics and Automation, San Diego, vol.1, pp.1608-1613, 1994.

[5] G. H. Golub, C.F. Van Loan: Matrix Computations, 3rd edition. Johns Hopkins University Press, 1996.

[6] F. L. Lewis: Neural Network Control of Robot Manipulators. IEEE Expert special track on Intelligent Control. vol.6, pp.64-75, 1996.

[7] A. Neumayer: Solving ill-conditioned and singular linear systems, a tutorial on regularization. SIAM Review, 40, pp. 636-666, 1998.

[8] Ch. Hansen: Rank Deficient and Discrete Ill Posed Problems, Numerical Aspects of Linear Inversion, SIAM, 1998

[9] P. I. Corke: Matlab Robotics Toolbox (release 5). CSIRO, Australia, 1999.

[10] F. L. Lewis: Neural Network Control of Robot Manipulators and Nonlinear Systems. Taylor & Fran-cis, 1999.

[11] H .J. Wesley, A. V. Gribok, A. M. Urmanov, M. A. Buckner: Selection of Multiple Regularization Parameters in Local Ridge Regression Using Evolutionary Algorithms and Prediction Risk Optimization. Inverse Problems in Engineering, vol. 11, num. 3, pp. 215-227(13), 2003.

[12] J. Możaryn, J. E. Kurek: Comparison of Neural Network Robot Models with Not Inverted and Inverted Inertia Matrix. Proc. International Conference on Artficial Neural Networks ICANN, Warsaw, 2005, vol.2, pp.417-422, 2005.

[13] J. Możaryn, J. E. Kurek: Improvement of Inertia Matrix in Robot Model Identified with Neural Net-works. Proc. 13th IEEE Int. Conf. on Methods and Models in Automation and Robotics MMAR 2007, vol. 2, 2007

[14] S. Osowski: Sieci neuronowe do przetwarzania informacji. Oficyna Wydawnicza PW, 2006.

Artykuł recenzowany