INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 10(5):643-653, October, 2015.

ANN Training Method with a Small Number of Examples Used for Robots Control

E. Ciupan , F. Lungu, C. Ciupan

Emilia Ciupan, Florin Lungu, Cornel Ciupan*
Technical University of Cluj-Napoca
Romania, 400641 Cluj-Napoca, Bd. Muncii, 103-105
emilia.ciupan@mis.utcluj.ro, florin.lungu@mis.utcluj.ro
*Corresponding author: cornel.ciupan@muri.utcluj.ro

Abstract: This paper presents a method for obtaining a neural model used in industrial robots control. The method refers to the forming of a small number of examples used in the training of a neural network that lead to the creation of a suitable model. This paper constitutes a development of the work [2] in order to increase the opportunities for its application in various fields. The description of the method is generally done, without relying on a specific application in the domain of industrial robots. The testing and the validation of the shown method were completed using the example of a system in which the relationship between inputs and outputs is described by means of mathematical functions. The set of learning examples, generated through the proposed method, served to the ANN training by a cross-validation technique, in case of these functions. The evaluation of the proposed method has been done by analysing the results obtained by applying it compared to those obtained with a known method, namely the uniform generation of training examples. The use of the method in the field of industrial robots' control was illustrated by a concrete application in the case of a robot with 6 degrees of freedom.
Keywords: ANN, training, method, small number, robot, control.

1 Introduction

The control proper is achieved by the robot's control equipment by generating a control value for each joint, so that the joint achieve coordinate q_i resulted from the inverse kinematics, and the effector move through points of coordinates that belong to the trajectory. Therefore determining the coordinates of the joints is of capital importance [7].

Neural networks can perform complex learning and adaptation tasks by imitating the function of biological neural systems, and thus can be used as models for nonlinear, multi-variable systems, trained by using input-output data observed on the system [3].

The application of neural networks to robots control is well known [1,4,6] and an alternative to the adaptive control is represented by the neural controllers [8].

The models based on neural networks show an advantage in terms of model simplification and, first and foremost, of the operations performed, as they consist solely of multiplications and additions. A model based on neural networks that would answer parallel robot control involves appropriate modelling and very good network training. The large number of the training data necessary for a high-quality neural model, which can reach thousands of examples [5], can often be problematic. That is why the problem that this paper solves consists in the completion of a method for obtaining a high-quality neural network with a small number of training examples, one that grants both the desired precision in the entire robot's workspace and a reduction of the training time. Let us consider a robot with six degrees of freedom having three translation axes that give the positioning movement X, Y, Z, and other three rotation axes that give the orientation movement ψ, θ, φ.

Based on the robot's kinematic scheme [4], there has been determined the mathematical model for the direct kinematics (1)-(2), as well as the model for inverse kinematic analysis (3)-(4):

$$X = q_1 \cdot i_{Tx} + X_0; Y = q_2 \cdot i_{Ty} + Y_0; Z = q_3 \cdot i_{Tz} + Z_0; \tag{1}$$

$$\psi = q_4 \cdot i_{T\psi} + \psi_0; \theta = q_5 \cdot i_{T\theta} + \theta_0; \varphi = q_6 \cdot i_{T\varphi} + \varphi_0; \tag{2}$$

$$q_1 = \frac{X - X_0}{i_{Tx}}; q_2 = \frac{Y - Y_0}{i_{Ty}}; q_3 = \frac{Z - Z_0}{i_{Tz}}; \tag{3}$$

$$q_4 = \frac{\psi - \psi_0}{i_{T\psi}}; q_5 = \frac{\theta - \theta_0}{i_{T\theta}}; q_6 = \frac{\varphi - \varphi_0}{i_{T\varphi}}; \tag{4}$$

where i_{Tx}, i_{Ty}, i_{Tz}, $i_{T\psi}$, $i_{T\theta}$, $i_{T\varphi}$ represent the transfer functions of the transformation mechanisms that generate the given movements and X_0, Y_0, Z_0, ψ_0, θ_0, φ_0 represent the values of the system origin for which q_i=0, i=1,...,6.

2 Description of the method used for the generation of training examples and its validation

The majority of robots have six degrees of freedom and, consequently, six joints. The current paper aims at finding a method of reducing the number of training examples of a neural network for the control of a robot with six degrees of freedom. In order to describe the method, a random system with 6 input signals, and 6 output ones, is being considered. In the process of building the neural network that is to shape the system, there is used a set of training examples in which the input is represented by vectors of values (q_1, q_2, q_3, q_4, q_5, q_6), where $q_i \in [q_{min}, q_{max}]$, $\forall i = \overline{1,6}$. The output values of the system are described by the values of the vectorial function $F = (f_1, f_2, f_3, f_4, f_5, f_6)$, where F:$[q_{min}, q_{max}]^6 \to R^6$.

According to the method proposed, the set of training examples results from the imposition of a successive move of the signals q_i, i=1,...,6, by the successive move of each signal by a step p, followed by mixed moves of several axes, for each set j of input data $q_{i,j}$, i=1,...,6 and j=1,...,m, where m is the total number of sets of this type, resulting in a set of output data $F(q_{1j}, q_{2j}, q_{3j}, q_{4j}, q_{5j}, q_{6j})$ that is used for the network training. The training and testing data is determined according to the mathematical model for the function F by modifying the input signals according to Table 1.

In the mathematical model for function F, sets of the input signals (q_1, q_2, q_3, q_4, q_5, q_6) are gradually being introduced, and the output values are calculated. The data obtained after the completion of Table 1 represents the set of training examples. The novelty of this method consists in the way used to determine the neural network training data in order to achieve a high-quality neural network by means of a small number of training examples. A high-quality neural network is defined here as a neural model that offers acceptable errors in the entire domain of function F. In order to train the network effectively with a relatively reduced number of training examples, we suggest the use of an algorithm of generating the set of training examples which implies the completion of 30 phases expressed concisely in Table 1. A phase, denoted here by h, corresponds to a set of pairs of six (q_i, i=1,...,6), one for each factor z of multiplication of the step p, as it is shown explicitly in phase 1 in the Table 1. The modification of the inputs is done based on the relation below:

Phase	Input values (q_i, i=1,…,6)						Output values $P_{h,z}(f_1,f_2,f_3,f_4,f_5,f_6)$, h=1,…,30, z=0,…,j					
h	q_1	q_2	q_3	q_4	q_5	q_6	f_1	f_2	f_3	f_4	f_5	f_6
1.	$q_{1min}+z\cdot p$	$q_{2\,min}$	$q_{3\,min}$	$q_{4\,min}$	$q_{5\,min}$	$q_{6\,min}$	$f1_{1:0}$	$f2_{1:0}$	$f3_{1:0}$	$f4_{1:0}$	$f5_{1:0}$	$f6_{1:0}$
							...	...	...	...	...	...
							$f1_{1:z}$	$f2_{1:z}$	$f3_{1:z}$	$f4_{1:z}$	$f5_{1:z}$	$f6_{1:z}$
							...	...	...	...	...	...
							$f1_{1:j}$	$f2_{1:j}$	$f3_{1:j}$	$f4_{1:j}$	$f5_{1:j}$	$f6_{1:j}$
2.	q_{1min}	$q_{2\,min}+z\cdot p$	$q_{3\,min}$	$q_{4\,min}$	$q_{5\,min}$	$q_{6\,min}$	$f1_{2:z}$	$f2_{2:z}$	$f3_{2:z}$	$f4_{2:z}$	$f5_{2:z}$	$f6_{2:z}$
3.	q_{1min}	$q_{2\,min}$	$q_{3\,min}+z\cdot p$	$q_{4\,min}$	$q_{5\,min}$	$q_{6\,min}$	$f1_{3:z}$	$f2_{3:z}$	$f3_{3:z}$	$f4_{3:z}$	$f5_{3:z}$	$f6_{3:z}$
4.	q_{1min}	$q_{2\,min}$	$q_{3\,min}$	$q_{4\,min}+z\cdot p$	$q_{5\,min}$	$q_{6\,min}$	$f1_{4:z}$	$f2_{4:z}$	$f3_{4:z}$	$f4_{4:z}$	$f5_{4:z}$	$f6_{4:z}$
5.	q_{1min}	$q_{2\,min}$	$q_{3\,min}$	$q_{4\,min}$	$q_{5\,min}+z\cdot p$	$q_{6\,min}$	$f1_{5:z}$	$f2_{5:z}$	$f3_{5:z}$	$f4_{5:z}$	$f5_{5:z}$	$f6_{5:z}$
6.	q_{1min}	$q_{2\,min}$	$q_{3\,min}$	$q_{4\,min}$	$q_{5\,min}$	$q_{6\,min}+z\cdot p$	$f1_{6:z}$	$f2_{6:z}$	$f3_{6:z}$	$f4_{6:z}$	$f5_{6:z}$	$f6_{6:z}$
7.	$q_{1min}+z\cdot p$	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,med}$	$f1_{7:z}$	$f2_{7:z}$	$f3_{7:z}$	$f4_{7:z}$	$f5_{7:z}$	$f6_{7:z}$
8.	q_{1med}	$q_{2\,min}+z\cdot p$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,med}$	$f1_{8:z}$	$f2_{8:z}$	$f3_{8:z}$	$f4_{8:z}$	$f5_{8:z}$	$f6_{8:z}$
9.	q_{1med}	$q_{2\,med}$	$q_{3\,min}+z\cdot p$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,med}$	$f1_{9:z}$	$f2_{9:z}$	$f3_{9:z}$	$f4_{9:z}$	$f5_{9:z}$	$f6_{9:z}$
10.	q_{1med}	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,min}+z\cdot p$	$q_{5\,med}$	$q_{6\,med}$	$f1_{10:z}$	$f2_{10:z}$	$f3_{10:z}$	$f4_{10:z}$	$f5_{10:z}$	$f6_{10:z}$
11.	q_{1med}	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,min}+z\cdot p$	$q_{6\,med}$	$f1_{11:z}$	$f2_{11:z}$	$f3_{11:z}$	$f4_{11:z}$	$f5_{11:z}$	$f6_{11:z}$
12.	q_{1med}	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,min}+z\cdot p$	$f1_{12:z}$	$f2_{12:z}$	$f3_{12:z}$	$f4_{12:z}$	$f5_{12:z}$	$f6_{12:z}$
13.	$q_{1min}+z\cdot p$	$q_{2\,max}$	$q_{3\,max}$	$q_{4\,max}$	$q_{5\,max}$	$q_{6\,max}$	$f1_{13:z}$	$f2_{13:z}$	$f3_{13:z}$	$f4_{13:z}$	$f5_{13:z}$	$f6_{13:z}$
14.	q_{1max}	$q_{2\,min}+z\cdot p$	$q_{3\,max}$	$q_{4\,max}$	$q_{5\,max}$	$q_{6\,max}$	$f1_{14:z}$	$f2_{14:z}$	$f3_{14:z}$	$f4_{14:z}$	$f5_{14:z}$	$f6_{14:z}$
15.	q_{1max}	$q_{2\,max}$	$q_{3\,min}+z\cdot p$	$q_{4\,max}$	$q_{5\,max}$	$q_{6\,max}$	$f1_{15:z}$	$f2_{15:z}$	$f3_{15:z}$	$f4_{15:z}$	$f5_{15:z}$	$f6_{15:z}$
16.	q_{1max}	$q_{2\,max}$	$q_{3\,max}$	$q_{4\,min}+z\cdot p$	$q_{5\,max}$	$q_{6\,max}$	$f1_{16:z}$	$f2_{16:z}$	$f3_{16:z}$	$f4_{16:z}$	$f5_{16:z}$	$f6_{16:z}$
17.	q_{1max}	$q_{2\,max}$	$q_{3\,max}$	$q_{4\,max}$	$q_{5\,min}+z\cdot p$	$q_{6\,max}$	$f1_{17:z}$	$f2_{17:z}$	$f3_{17:z}$	$f4_{17:z}$	$f5_{17:z}$	$f6_{17:z}$
18.	q_{1max}	$q_{2\,max}$	$q_{3\,max}$	$q_{4\,max}$	$q_{5\,max}$	$q_{6\,min}+z\cdot p$	$f1_{18:z}$	$f2_{18:z}$	$f3_{18:z}$	$f4_{18:z}$	$f5_{18:z}$	$f6_{18:z}$
19.	$q_{1min}+z\cdot p$	$q_{2\,min}+z\cdot p$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,med}$	$f1_{19:z}$	$f2_{19:z}$	$f3_{19:z}$	$f4_{19:z}$	$f5_{19:z}$	$f6_{19:z}$
20.	q_{1med}	$q_{2\,med}$	$q_{3\,min}+z\cdot p$	$q_{4\,min}+z\cdot p$	$q_{5\,med}$	$q_{6\,med}$	$f1_{20:z}$	$f2_{20:z}$	$f3_{20:z}$	$f4_{20:z}$	$f5_{20:z}$	$f6_{20:z}$
21.	q_{1med}	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,med}$	$q_{5\,min}+z\cdot p$	$q_{6\,min}+z\cdot p$	$f1_{21:z}$	$f2_{21:z}$	$f3_{21:z}$	$f4_{21:z}$	$f5_{21:z}$	$f6_{21:z}$
22.	$q_{1min}+z\cdot p$	$q_{2\,min}+z\cdot p$	$q_{3\,min}+z\cdot p$	$q_{4\,med}$	$q_{5\,med}$	$q_{6\,med}$	$f1_{22:z}$	$f2_{22:z}$	$f3_{22:z}$	$f4_{22:z}$	$f5_{22:z}$	$f6_{22:z}$
23.	q_{1med}	$q_{2\,med}$	$q_{3\,med}$	$q_{4\,min}+z\cdot p$	$q_{5\,min}+z\cdot p$	$q_{6\,min}+z\cdot p$	$f1_{23:z}$	$f2_{23:z}$	$f3_{23:z}$	$f4_{23:z}$	$f5_{23:z}$	$f6_{23:z}$
24.	$q_{1min}+z\cdot p$	$q_{2\,min}+z\cdot p$	$q_{3\,min}+z\cdot p$	$q_{4\,min}+z\cdot p$	$q_{5\,med}$	$q_{6\,med}$	$f1_{24:z}$	$f2_{24:z}$	$f3_{24:z}$	$f4_{24:z}$	$f5_{24:z}$	$f6_{24:z}$
25.	q_{1med}	$q_{2\,med}$	$q_{3\,min}+z\cdot p$	$q_{4\,min}+z\cdot p$	$q_{5\,min}+z\cdot p$	$q_{6\,min}+z\cdot p$	$f1_{25:z}$	$f2_{25:z}$	$f3_{25:z}$	$f4_{25:z}$	$f5_{25:z}$	$f6_{25:z}$
26.	$q_{1min}+z\cdot p$	$q_{2\,min}+z\cdot p$	$q_{3\,min}+z\cdot p$	$q_{4\,min}+z\cdot p$	$q_{5\,min}+z\cdot p$	$q_{6\,min}+z\cdot p$	$f1_{26:z}$	$f2_{26:z}$	$f3_{26:z}$	$f4_{26:z}$	$f5_{26:z}$	$f6_{26:z}$
27.	$q_{1min}+z\cdot p$	$q_{2\,min}$	$q_{3\,min}+z\cdot p$	$q_{4\,min}$	$q_{5\,min}+z\cdot p$	$q_{6\,min}$	$f1_{27:z}$	$f2_{27:z}$	$f3_{27:z}$	$f4_{27:z}$	$f5_{27:z}$	$f6_{27:z}$
28.	$q_{1min}+z\cdot p$	$q_{2\,max}$	$q_{3\,min}+z\cdot p$	$q_{4\,max}$	$q_{5\,min}+z\cdot p$	$q_{6\,max}$	$f1_{28:z}$	$f2_{28:z}$	$f3_{28:z}$	$f4_{28:z}$	$f5_{28:z}$	$f6_{28:z}$
29.	q_{1min}	$q_{2\,min}+z\cdot p$	$q_{3\,min}$	$q_{4\,min}+z\cdot p$	$q_{5\,min}$	$q_{6\,min}+z\cdot p$	$f1_{29:z}$	$f2_{29:z}$	$f3_{29:z}$	$f4_{29:z}$	$f5_{29:z}$	$f6_{29:z}$
30.	q_{1max}	$q_{2\,min}+z\cdot p$	$q_{3\,max}$	$q_{4\,min}+z\cdot p$	$q_{5\,max}$	$q_{6\,min}+z\cdot p$	$f1_{30:z}$	$f2_{30:z}$	$f3_{30:z}$	$f4_{30:z}$	$f5_{30:z}$	$f6_{30:z}$

Table 1: Generation of training examples

$$q_i = q_{imin} + z \cdot p; \tag{5}$$

where z=0,...,j for all the 30 phases, except phases 2-6, where z=1,...,j and p represents the increase step given by the relation:

$$p = \frac{q_{imax} - q_{imin}}{j}; \tag{6}$$

and j is the number of steps. For the validation of the method, there has been considered the example in which the input signals $q_i \in [0, 90]$, $\forall i = \overline{1, 6}$, the step p=5, and the components of the vectorial function F are defined by the following equations:

$$f_1(q_1, q_2, q_3, q_4, q_5, q_6) = 10 \cdot sinq_1 \tag{7}$$

$$f_2(q_1, q_2, q_3, q_4, q_5, q_6) = 10 \cdot (1 - cosq_2) \tag{8}$$

$$f_3(q_1, q_2, q_3, q_4, q_5, q_6) = \frac{10}{q^2_{max}} \cdot q^2_3 \tag{9}$$

$$f_4(q_1, q_2, q_3, q_4, q_5, q_6) = \frac{q_4}{9} \tag{10}$$

$$f_5(q_1, q_2, q_3, q_4, q_5, q_6) = \frac{q_5}{9} \tag{11}$$

$$f_6(q_1, q_2, q_3, q_4, q_5, q_6) = \frac{q_6}{9} \tag{12}$$

According to the rule described concisely in Table 1, there has been generated a set of training data of the neural network formed by pairs of examples $((q_1, q_2, q_3, q_4, q_5, q_6), F(q_1, q_2, q_3, q_4, q_5, q_6))$. After the elimination of the repeated values, there has been obtained a set of 382 examples, marked as trainProp. In order to compare the method proposed with the known method, there has been generated the set trainKnown, with a number of elements (375) close to that of the set trainProp. The training examples of this set have been obtained by means of the same approach as in the case of the robots inverse kinematic analysis. The starting point was made of uniformely distributed values (step 3.3) in the image of function F, respectively $[0, 10]^6$. The values of the corresponding input signals q_i, i=1,...,6 has been determined by means of the inverses of the functions $f_1, ..., f_6$. Table 2 illustrates the two sets, trainProp and trainKnown.

In the case of both sets, there has been applied the technique 4-fold cross-validation for the random division of each initial set of training examples into four subsets $sSetProp_l$, and $sSetKnown_l$, $l = \overline{1, 4}$, respectively. The subsets of each set have been used, in a combination of three, to the training of the network, while the fourth subset has been used for the validation of the model throughout the training process. We mark the neural models obtained as a result of the instruction as $netProp_k$ and $netKnown_k$,$k = \overline{1, 4}$, respectively. The errors obtained in the training process are close in the case of the two methods used, and they are of order 10^{-7}. In order to test the netProp$_k$ and netKnown$_k$ models, k=$\overline{1, 4}$, there has been constituted the set of input-output data rendered in Table 3.

The data in Table 4 show better results in the cases in which, for the training stage, there has been used the set obtained by means of the method proposed, as compared to the set achieved by means of the method known. More detailed data concerning the individual errors of the output signals are comprised in the tables 5 and 6, both in the case of the method proposed, and in that of the method known. The variations of the absolute errors of the output signals f_1, f_2 and f_3 are shown in the graphs that belong to Table 5, while those of the signals f_4, f_5 and f_6, are captured in the graphs corresponding to Table 6.

It can be noticed the lesser size order of the individual errors in the case of the network training by means of data obtained through the method proposed, as compared to the one known.

Proposed method												
No.	Input values (q_i, i=1,...,6)						Output values F(q_i, i=1,...,6)					
	q1	q2	q3	q4	q5	q6	f1	f2	f3	f4	f5	f6
1	0	0	0	0	0	0	0.00	0	0	0	0	0
2	5	0	0	0	0	0	0.87	0	0	0	0	0
3	10	0	0	0	0	0	1.74	0	0	0	0	0
4	15	0	0	0	0	0	2.59	0	0	0	0	0
5	20	0	0	0	0	0	3.42	0	0	0	0	0
6	25	0	0	0	0	0	4.23	0	0	0	0	0
7	30	0	0	0	0	0	5.00	0	0	0	0	0
8	35	0	0	0	0	0	5.74	0	0	0	0	0
9	40	0	0	0	0	0	6.43	0	0	0	0	0
10	45	0	0	0	0	0	7.07	0	0	0	0	0
11	50	0	0	0	0	0	7.66	0	0	0	0	0
12	55	0	0	0	0	0	8.19	0	0	0	0	0
13	60	0	0	0	0	0	8.66	0	0	0	0	0
14	65	0	0	0	0	0	9.06	0	0	0	0	0
15	70	0	0	0	0	0	9.40	0	0	0	0	0
16	75	0	0	0	0	0	9.66	0	0	0	0	0
17	80	0	0	0	0	0	9.85	0	0	0	0	0
18	85	0	0	0	0	0	9.96	0	0	0	0	0
19	90	0	0	0	0	0	10	0	0	0	0	0
...												
365	90	5	90	5	90	5	10	0.04	10	0.56	10	0.56
366	90	10	90	10	90	10	10	0.15	10	1.11	10	1.11
367	90	15	90	15	90	15	10	0.34	10	1.67	10	1.67
368	90	20	90	20	90	20	10	0.60	10	2.22	10	2.22
369	90	25	90	25	90	25	10	0.94	10	2.78	10	2.78
370	90	30	90	30	90	30	10	1.34	10	3.33	10	3.33
371	90	35	90	35	90	35	10	1.81	10	3.89	10	3.89
372	90	40	90	40	90	40	10	2.34	10	4.44	10	4.44
373	90	45	90	45	90	45	10	2.93	10	5.00	10	5.00
374	90	50	90	50	90	50	10	3.57	10	5.56	10	5.56
375	90	55	90	55	90	55	10	4.26	10	6.11	10	6.11
376	90	60	90	60	90	60	10	5.00	10	6.67	10	6.67
377	90	65	90	65	90	65	10	5.77	10	7.22	10	7.22
378	90	70	90	70	90	70	10	6.58	10	7.78	10	7.78
379	90	75	90	75	90	75	10	7.41	10	8.33	10	8.33
380	90	80	90	80	90	80	10	8.26	10	8.89	10	8.89
381	90	85	90	85	90	85	10	9.13	10	9.44	10	9.44
382	90	90	90	90	90	90	10	10	10	10	10	10

Known method												
No.	Input values (q_i, i=1,...,6)						Output values F(qi, i=1,...,6)					
	q1	q2	q3	q4	q5	q6	f1	f2	f3	f4	f5	f6
1	0.00	47.93	51.70	29.70	29.70	29.70	0	3.3	3.3	3.3	3.3	3.3
2	19.27	47.93	51.70	29.70	29.70	29.70	3.3	3.3	3.3	3.3	3.3	3.3
3	41.30	47.93	51.70	29.70	29.70	29.70	6.6	3.3	3.3	3.3	3.3	3.3
4	90.00	47.93	51.70	29.70	29.70	29.70	10	3.3	3.3	3.3	3.3	3.3
5	19.27	0.00	51.70	29.70	29.70	29.70	3.3	0	3.3	3.3	3.3	3.3
6	19.27	70.12	51.70	29.70	29.70	29.70	3.3	6.6	3.3	3.3	3.3	3.3
7	19.27	90.00	51.70	29.70	29.70	29.70	3.3	10	3.3	3.3	3.3	3.3
8	19.27	47.93	0.00	29.70	29.70	29.70	3.3	3.3	0	3.3	3.3	3.3
9	19.27	47.93	73.12	29.70	29.70	29.70	3.3	3.3	6.6	3.3	3.3	3.3
10	19.27	47.93	90.00	29.70	29.70	29.70	3.3	3.3	10	3.3	3.3	3.3
11	19.27	47.93	51.70	0.00	29.70	29.70	3.3	3.3	3.3	0	3.3	3.3
12	19.27	47.93	51.70	59.40	29.70	29.70	3.3	3.3	3.3	6.6	3.3	3.3
13	19.27	47.93	51.70	90.00	29.70	29.70	3.3	3.3	3.3	10	3.3	3.3
14	19.27	47.93	51.70	29.70	0.00	29.70	3.3	3.3	3.3	3.3	0	3.3
15	19.27	47.93	51.70	29.70	59.40	29.70	3.3	3.3	3.3	3.3	6.6	3.3
16	19.27	47.93	51.70	29.70	90.00	29.70	3.3	3.3	3.3	3.3	10	3.3
17	19.27	47.93	51.70	29.70	29.70	0.00	3.3	3.3	3.3	3.3	3.3	0
18	19.27	47.93	51.70	29.70	29.70	59.40	3.3	3.3	3.3	3.3	3.3	6.6
19	19.27	47.93	51.70	29.70	29.70	90.00	3.3	3.3	3.3	3.3	3.3	10
...												
358	19.27	90.00	90.00	90.00	90.00	90.00	3.3	10	10	10	10	10
359	41.30	90.00	90.00	90.00	90.00	90.00	6.6	10	10	10	10	10
360	90.00	0.00	90.00	90.00	90.00	90.00	10	0	10	10	10	10
361	90.00	47.93	90.00	90.00	90.00	90.00	10	3.3	10	10	10	10
362	90.00	70.12	90.00	90.00	90.00	90.00	10	6.6	10	10	10	10
363	90.00	90.00	0.00	90.00	90.00	90.00	10	10	0	10	10	10
364	90.00	90.00	51.70	90.00	90.00	90.00	10	10	3.3	10	10	10
365	90.00	90.00	73.12	90.00	90.00	90.00	10	10	6.6	10	10	10
366	90.00	90.00	90.00	0.00	90.00	90.00	10	10	10	0	10	10
367	90.00	90.00	90.00	29.70	90.00	90.00	10	10	10	3.3	10	10
368	90.00	90.00	90.00	59.40	90.00	90.00	10	10	10	6.6	10	10
369	90.00	90.00	90.00	90.00	0.00	90.00	10	10	10	10	0	10
370	90.00	90.00	90.00	90.00	29.70	90.00	10	10	10	10	3.3	10
371	90.00	90.00	90.00	90.00	59.40	90.00	10	10	10	10	6.6	10
372	90.00	90.00	90.00	90.00	90.00	0.00	10	10	10	10	10	0
373	90.00	90.00	90.00	90.00	90.00	29.70	10	10	10	10	10	3.3
374	90.00	90.00	90.00	90.00	90.00	59.40	10	10	10	10	10	6.6
375	90.00	90.00	90.00	90.00	90.00	90.00	10	10	10	10	10	10

Table 2: Training data generation

No	Inputs						Outputs					
	q1	q2	q3	q4	q5	q6	f1	f2	f3	f4	f5	f6
1	0.5	0.5	0.5	0.5	0.5	0.5	0.087265	0.000381	0.000309	0.055556	0.055556	0.055556
2	1.5	1.5	1.5	1.5	1.5	1.5	0.261769	0.003427	0.002778	0.166667	0.166667	0.166667
3	2.5	2.5	2.5	2.5	2.5	2.5	0.436194	0.009518	0.007716	0.277778	0.277778	0.277778
4	3.5	3.5	3.5	3.5	3.5	3.5	0.610485	0.018652	0.015123	0.388889	0.388889	0.388889
5	4.5	4.5	4.5	4.5	4.5	4.5	0.784591	0.030827	0.025	0.5	0.5	0.5
...												
86	85.5	85.5	85.5	85.5	85.5	85.5	9.969173	9.215409	9.025	9.5	9.5	9.5
87	86.5	86.5	86.5	86.5	86.5	86.5	9.981348	9.389515	9.237346	9.611111	9.611111	9.611111
88	87.5	87.5	87.5	87.5	87.5	87.5	9.990482	9.563806	9.45216	9.722222	9.722222	9.722222
89	88.5	88.5	88.5	88.5	88.5	88.5	9.996573	9.738231	9.669444	9.833333	9.833333	9.833333
90	89.5	89.5	89.5	89.5	89.5	89.5	9.999619	9.912735	9.889198	9.944444	9.944444	9.944444

Table 3: Testing data

k	Mean square error	
	netProp$_k$	netKnown$_k$
1	2.25E-04	8.37E-02
2	1.83E-04	8.34E-02
3	2.39E-04	8.45E-02
4	1.93E-04	7.88E-02

Table 4: Testing results

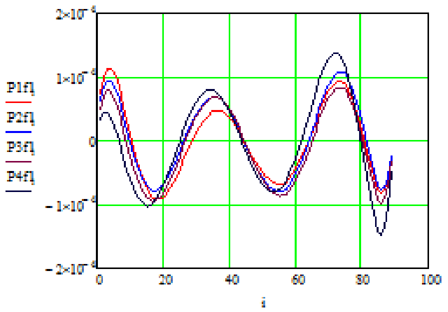
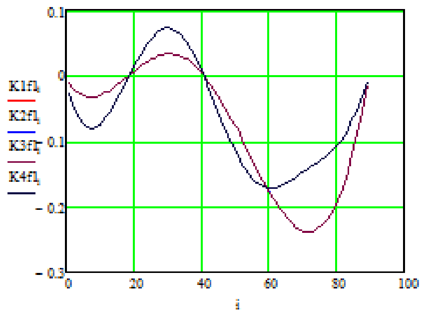
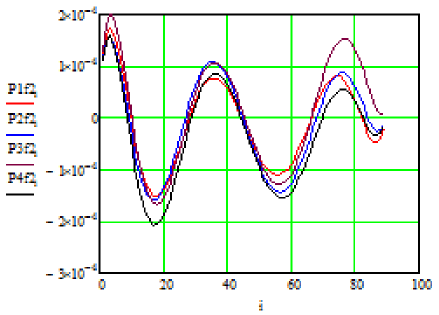
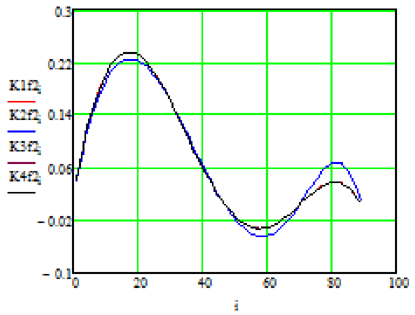
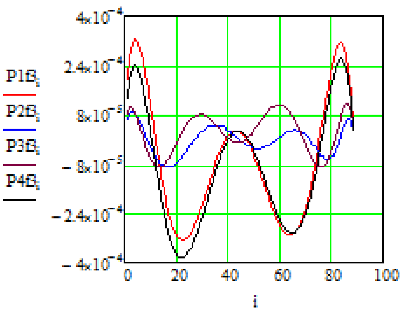
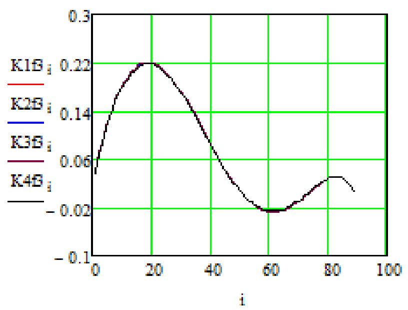
Output signal	Method proposed	Method known
f1		
f2		
f3		

Table 5: The error variation in the case of the signals f_1, f_2, f_3

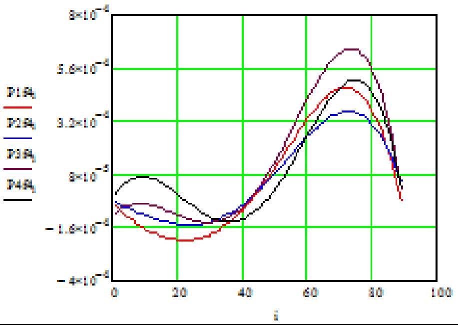
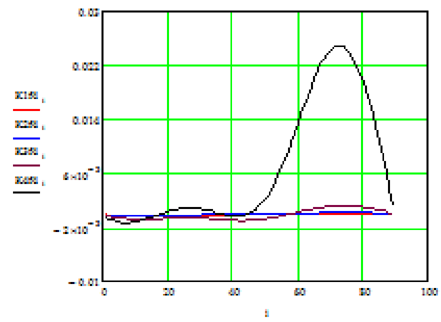
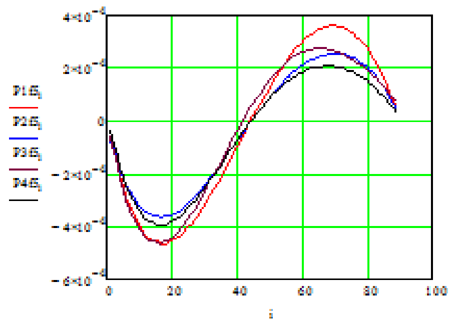
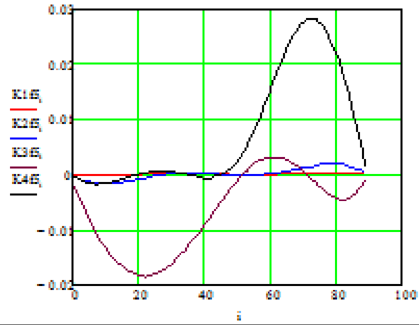
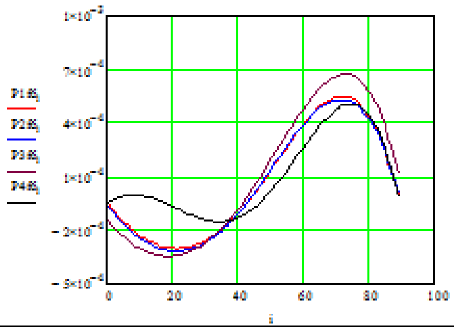
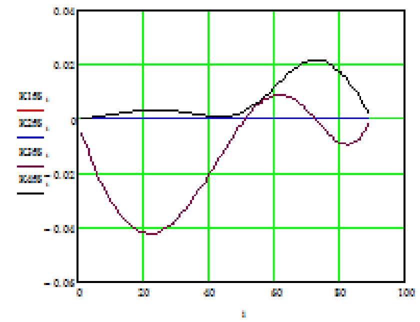
Output signal	Method proposed	Method known
f4		
f5		
f6		

Table 6: The error variation in the case of the signals f_4, f_5, f_6

3 Application for a robot with 6 degrees of freedom

Let us consider the robots workspace as a cube with the side of 300 mm. Based on the algorithm shown in Table 1, there will be generated three sets of training examples for the method proposed here, taking into account in the model for direct kinematic analysis (1)-(2) the following constants:

$$i_{Tx} = i_{Ty} = i_{Tz} = 10; i_{T\psi} = i_{T\theta} = i_{T\varphi} = 3; \tag{13}$$

$$X_0 = 0; Y_0 = 0; Z_0 = 300; \psi_0 = \theta_0 = \varphi_0 = -60^o; \tag{14}$$

The above value of -60^o for the angular variables was chosen so that the coordinates of the joints differs from 0 in the system reference point.

In order to validate the method suggested here, an alternative is considered to be the method of formation the training examples by the evenly distributed choice of the coordinates of the effector in the robot's workspace. For the method known, of evenly distributed choice of the coordinates, there will be generated three sets of data. The training data is obtained from the model for inverse kinematic analysis (3)-(4) with initial values from relation (13)-(14). In order to generate each set of training data (Set 1, Set 2, Set 3) corresponding to the two methods, there has been used the data in Table 7.

Data set		q_1	q_2	q_3	q_4	q_5	q_6	No. of examples n
Set 1	q_{imin}	20	20	20	20	20	20	n=104
	q_{imax}	50	50	50	50	50	50	
	p	10	10	10	10	10	10	
	j	3	3	3	3	3	3	
Set 2	q_{imin}	20	20	20	20	20	20	n=194
	q_{imax}	50	50	50	50	50	50	
	p	5	5	5	5	5	5	
	j	6	6	6	6	6	6	
Set 3	q_{imin}	20	20	20	20	20	20	n=448
	q_{imax}	50	50	50	50	50	50	
	p	2	2	2	2	2	2	
	j	15	15	15	15	15	15	
		X	Y	Z	ψ	θ	φ	
	min	200	200	200	0	0	0	
	max	500	500	500	90	90	90	

Data set	X	Y	Z	ψ	θ	φ	No. of examples n
Set 1	200 350 500	200 350 500	500 650 800	0	0	0	n=3x3x3x4=108
				90	0	0	
				0	90	0	
				0	0	90	
Set 2	200	200	200	0	0	0	n=4x4x4x4=256
	300	300	300	90	0	0	
	400	400	400	0	90	0	
	500	500	500	0	0	90	
Set 3	200 300 400 500	200 300 400 500	500 600 700 800	0	0	0	n=4x4x4x6=448
				45	0	0	
				0	45	0	
				0	0	45	
				90	0	0	
				0	90	0	
				0	0	90	

Table 7: Training data sets

The number of a set's training examples was denoted by n and it acquired different values in the case of each set. Using the data in Tables 1 and 7 there has been generated for each method a set of training data. In order to make a comparison between the results of the two methods, there have been used sets of data having the cardinality of the set of training examples (n) close (Set 1, Set 2) or identical (Set 3).

Table 8 shows in parallel the set of training data for the proposed method, and Set 1 for the known method. There has been modelled a three-layer neural network (6-20-6) and the training with the three sets of data resulted in three neural models for each method. For each set of data and for each method, there have been obtained close errors of training ($e < 10^{-5}$). In order to compare the two methods, the neural networks have been tested for each set and method. For the comparison of the results, there has been taken into account the move of the effector in the robot's workspace on a diagonal between the coordinate points P_1(210, 210, 670) and P_2(490, 490, 790). For the orientation of the effector, there has been considered a rotation $\varphi = 90^o$. The values of the input signals are shown in columns 1-6 in Table 9. The positioning error in Table 9 represents the resultant of the positioning coordinates error (X, Y, Z). Similarly, the angular error represents the resultant of the angular errors (ψ, θ, φ). Table 10 shows graphically the

Proposed method

No.	Stage	Joints coordinates ($q_{i,n}$, i=1,...,6; n=1,...,104)						Effector coordinates P_n(X,Y,Z, ψ, θ, φ), n=1,...,104					
t	f	q_1	q_2	q_3	q_4	q_5	q_6	X	Y	Z	ψ	θ	φ
1	1	20	20	20	20	20	20	200	200	500	0	0	0
2		30	20	20	20	20	20	300	200	500	0	0	0
3		40	20	20	20	20	20	400	200	500	0	0	0
4		50	20	20	20	20	20	500	200	500	0	0	0
5	2	20	30	20	20	20	20	200	300	500	0	0	0
6		20	40	20	20	20	20	200	400	500	0	0	0
7		20	50	20	20	20	20	200	500	500	0	0	0
	...												
17	6	20	20	20	20	20	30	200	200	500	0	0	30
18		20	20	20	20	20	40	200	200	500	0	0	60
19		20	20	20	20	20	50	200	200	500	0	0	90
20	7	20	35	35	35	35	35	200	350	650	45	45	45
21		30	35	35	35	35	35	300	350	650	45	45	45
22		40	35	35	35	35	35	400	350	650	45	45	45
23		50	35	35	35	35	35	500	350	650	45	45	45
	...												
40	12	35	35	35	35	35	20	350	350	650	45	45	0
41		35	35	35	35	35	30	350	350	650	45	45	30
42		35	35	35	35	35	40	350	350	650	45	45	60
43		35	35	35	35	35	50	350	350	650	45	45	90
44	13	20	50	50	50	50	50	200	500	800	90	90	90
45		30	50	50	50	50	50	300	500	800	90	90	90
46		40	50	50	50	50	50	400	500	800	90	90	90
60	18	50	50	50	50	50	20	500	500	800	90	90	0
61		50	50	50	50	50	30	500	500	800	90	90	30
62		50	50	50	50	50	40	500	500	800	90	90	60
63	19	20	20	35	35	35	35	200	200	650	45	45	45
	...												
74	21	35	35	35	35	50	50	350	350	650	45	90	90
75	22	20	20	20	35	35	35	200	200	500	45	45	45
	...												
82	23	35	35	35	50	50	50	350	350	650	90	90	90
83	24	20	20	20	20	35	35	200	200	500	0	45	45
	...												
90	25	35	35	50	50	50	50	350	350	800	90	90	90
91	26	30	30	30	30	30	30	300	300	600	30	30	30
92		40	40	40	40	40	40	400	400	700	60	60	60
93	27	30	20	30	20	30	20	300	200	600	0	30	0
94		40	20	40	20	40	20	400	200	700	0	60	0
	...												
102	30	50	20	50	20	50	20	500	200	800	0	90	0
103		50	30	50	30	50	30	500	300	800	30	90	30
104		50	40	50	40	50	40	500	400	800	60	90	60

Known method

Stage	Joints coordinates ($q_{i,n}$, i=1,...,6; n=1,...,108)						Effector coordinates P_n(X,Y,Z, ψ, θ, φ), n=1,...,108					
f	q_1	q_2	q_3	q_4	q_5	q_6	X	Y	Z	ψ	θ	φ
1	20	20	20	20	20	20	200	200	500	0	0	0
2	35	20	20	20	20	20	350	200	500	0	0	0
3	50	20	20	20	20	20	500	200	500	0	0	0
4	20	35	20	20	20	20	200	350	500	0	0	0
5	35	35	20	20	20	20	350	350	500	0	0	0
6	50	35	20	20	20	20	500	350	500	0	0	0
7	20	50	20	20	20	20	200	500	500	0	0	0
8	35	50	20	20	20	20	350	500	500	0	0	0
9	50	50	20	20	20	20	500	500	500	0	0	0
10	20	20	35	20	20	20	200	200	650	0	0	0
...												
19	20	20	50	20	20	20	200	200	800	0	0	0
20	35	20	50	20	20	20	350	200	800	0	0	0
21	50	20	50	20	20	20	500	200	800	0	0	0
22	20	35	50	20	20	20	200	350	800	0	0	0
23	35	35	50	20	20	20	350	350	800	0	0	0
24	50	35	50	20	20	20	500	350	800	0	0	0
25	20	50	50	20	20	20	200	500	800	0	0	0
26	35	50	50	20	20	20	350	500	800	0	0	0
27	50	50	50	20	20	20	500	500	800	0	0	0
28	20	20	20	50	20	20	200	200	500	90	0	0
29	35	20	20	50	20	20	350	200	500	90	0	0
30	50	20	20	50	20	20	500	200	500	90	0	0
...												
51	50	35	50	50	20	20	500	350	800	90	0	0
52	20	50	50	50	20	20	200	500	800	90	0	0
53	35	50	50	50	20	20	350	500	800	90	0	0
54	50	50	50	50	20	20	500	500	800	90	0	0
55	20	20	20	20	50	20	200	200	500	0	90	0
56	35	20	20	20	50	20	350	200	500	0	90	0
57	50	20	20	20	50	20	500	200	500	0	90	0
...												
78	50	35	50	20	50	20	500	350	800	0	90	0
79	20	50	50	20	50	20	200	500	800	0	90	0
80	35	50	50	20	50	20	350	500	800	0	90	0
81	50	50	50	20	50	20	500	500	800	0	90	0
82	20	20	20	20	20	50	200	200	500	0	0	90
83	35	20	20	20	20	50	350	200	500	0	0	90
84	50	20	20	20	20	50	500	200	500	0	0	90
...												
105	50	35	50	20	20	50	500	350	800	0	0	90
106	20	50	50	20	20	50	200	500	800	0	0	90
107	35	50	50	20	20	50	350	500	800	0	0	90
108	50	50	50	20	20	50	500	500	800	0	0	90

Table 8: Set1 of training data for proposed and known method

Testing points	Effector coordinates						Proposed method (PrM)						Known method (KnM)					
							Set 1		Set 2		Set 3		Set 1		Set 2		Set 3	
Pn	P_n(X,Y,Z, ψ, θ, φ), n=1,...,11						Position Error [mm]	Angular Error [°]	Position Error [mm]	Angular Error [°]	Position Error [mm]	Angular Error [°]	Position Error [mm]	Angular Error [°]	Position Error [mm]	Angular Error [°]	Position Error [mm]	Angular Error [°]
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
n	X	Y	Z	ψ	θ	φ	0.0833	0.0000	0.0015	0.0000	0.0418	0.0000	1.4620	0.0000	0.1597	0.0000	0.0000	0.0000
1	210	210	510	0	0	0	0.2298	0.0020	0.0093	0.0002	0.0795	0.0028	18.6663	0.1243	6.0701	0.1582	0.0914	0.2883
2	230	230	530	0	0	6.4	0.3284	0.0023	0.0102	0.0007	0.0774	0.0060	31.9270	0.4877	8.5636	0.5283	0.0831	1.1327
3	250	250	550	0	0	12.88	0.4247	0.0017	0.0105	0.0009	0.0426	0.0039	39.0282	1.0491	7.7047	0.9262	0.0408	2.4159
4	270	270	570	0	0	19.32	0.5645	0.0038	0.0144	0.0009	0.0316	0.0047	39.9710	1.7607	5.9324	1.1810	0.0143	3.8805
5	290	290	590	0	0	25.76	0.7743	0.0118	0.0203	0.0010	0.0783	0.0125	35.3201	2.5621	9.6531	1.1573	0.0076	4.8799
6	310	310	610	0	0	32.2	1.0592	0.0265	0.0291	0.0014	0.0917	0.0083	26.4539	3.3790	17.6429	0.7891	0.0145	4.0579
7	330	330	630	0	0	38.64	1.4172	0.0459	0.0471	0.0029	0.0674	0.0115	15.9495	4.1216	26.2353	0.1009	0.0428	0.0715
8	350	350	650	0	0	45.08	1.8491	0.0650	0.0808	0.0060	0.0424	0.0425	10.3252	4.6882	33.4875	0.7905	0.0621	6.4541
9	370	370	670	0	0	51.52	2.3479	0.0769	0.1337	0.0116	0.0767	0.0759	15.6619	4.9747	38.0005	1.6983	0.0619	11.3875
10	390	390	690	0	0	57.96	2.8678	0.0746	0.2087	0.0201	0.1089	0.1033	22.0037	4.8883	38.8002	2.4044	0.0427	12.9225
11	410	410	710	0	0	64.4	3.3076	0.0545	0.3082	0.0322	0.1113	0.1215	24.4418	4.3602	35.3529	2.7036	0.0168	11.6250
12	430	430	730	0	0	70.84	3.5670	0.0221	0.4345	0.0482	0.0981	0.1319	21.6759	3.3542	27.5287	2.4402	0.0170	8.5784
13	450	450	750	0	0	77.28	3.6831	0.0015	0.5897	0.0682	0.1296	0.1390	13.4760	1.8705	15.5068	1.5280	0.0197	4.5242
14	470	470	770	0	0	83.72	3.8700	0.0394	0.7739	0.0919	0.2424	0.1476	1.4620	0.0005	0.1648	0.0002	0.0404	0.0000
15	490	490	790	0	0	90	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0832	0.0000
Standard deviation							1.4209	0.028563	0.2389	0.0282	0.0543	0.058182	12.8225	1.9209	14.2201	0.9163	0.0292	4.5026
Maximum error							3.8700	0.076854	0.7739	0.0919	0.2424	0.147576	39.9710	4.9747	38.8002	2.7036	0.0914	12.9225

Table 9: Simulated results

positioning and angular errors (represented in the graph as Pos. Er. and Ang. Er.) resulted from the simulation of the move between points P_1 and P_2. The results simulated with each neural model (Set 1, Set 2, Set 3) of each method has been marked graphically with PrM for the proposed method and KnM, for the known method.

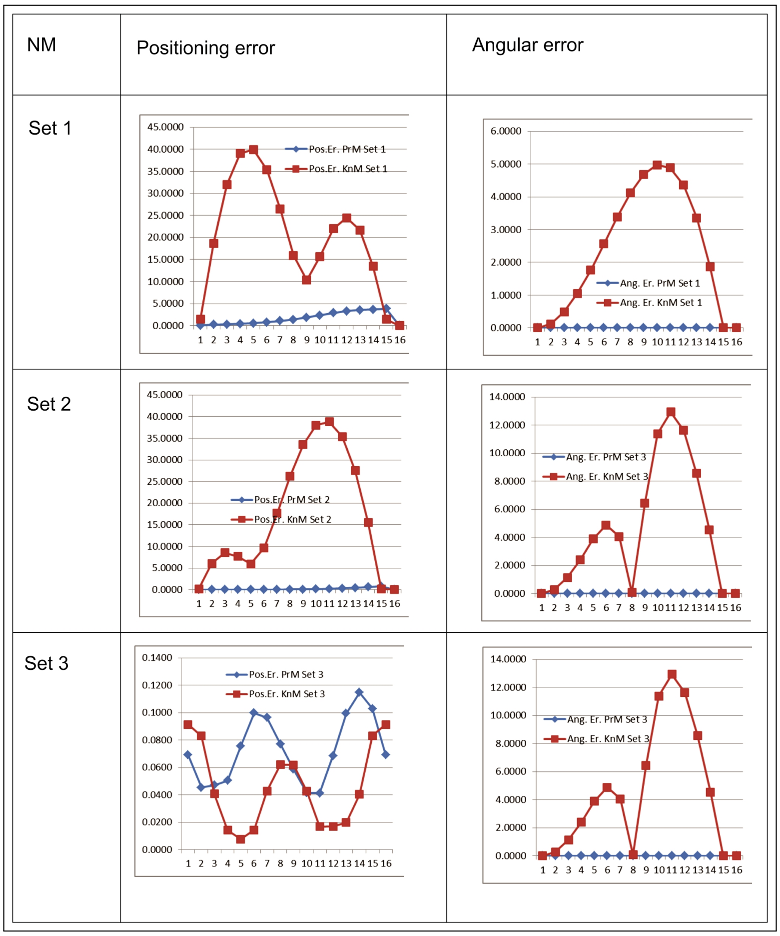


Table 10: Graphical representation of the errors resulted when simulating

4 Conclusions

The analysis of the results in Tables 9 and 10 in terms of positioning, angular and cumulated errors demonstrates the following:

- When simulating the move in a straight line, all the three neural models obtained based on the proposed method PrM offer much better results than the models obtained based on the KnM.

- The neural model Set 2-PrM offers positioning precision corresponding to some handling applications ($e < 1$ mm) although the set of training examples is relatively small (n=108). By contrast, even if for the neural model Set 2-KnM there has been used a relatively more significant number of training examples (n=256), this gives greater positioning errors ($e < 39$ mm) which cannot be accepted.

- The neural model Set 3-PrM (n=448) offers a positioning precision corresponding to the majority of the handling applications ($e < 0.25$ mm). The neural model Set 3-KnM (n=448) offers the best positioning precision ($e < 0.1$ mm), but it does not solve the orientation problem, the angular error being greater than 100.

It has been noted that the neural models generated by the proposed method offer better results as compared to the method of evenly distributed training data in the robot's workspace. The authors find that the proposed method offers superior results, and that it can be used in order to obtain high-quality neural results, with a reduced number of training data.

Bibliography

[1] Campean E.; Itul T.P.; Tanase I.; Pisla A.; Workspace Generation for a 2 - DOF Parallel Mechanism Using Neural Networks, *Applied Mechanics and Materials*, Vol 162 (2012), pp 121-130, doi: 10.4028/www.scientific.net/AMM.162.121.

[2] Ciupan E.; Lungu F.; Ciupan C. (2014); A Method for Training a Neural Network with a Small Number of Examples Used for Robot Control, *ICCCC 2014, 5th International Conference on Computers Communications & Control*, Romania, Oradea, May 6-10.

[3] Dumitrache, I. (2008); From Model-Based Strategies to Intelligent Control Systems, *Proc. of 9th WSEAS International Conference on Automation and Information*, 408-415.

[4] Feng Y.; Wanh Y.; Yang Y. (2012); Inverse Kinematics Solution for Robot Manipulator based on Neural Network under Joint Subspace, *International Journal of Computers Communications & Control*, 7(3):459-472.

[5] Koker R. (2013); A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, *Information Sciences*, 222, 528-543.

[6] Lewis F. L.; Jagannathan S.; Yesildirak A. (1998); *Neural network control for robots manipulators and nonlinear systems*, Tylor & Francis.

[7] Negrean I.; Vuscan I.; Haiduc N. (1998); *Kinematic and Dynamic Modelling*, Editura Didactica si Pedagogica, Bucuresti, ISBN 973-30-5958-7.

[8] Zilouchian A.; Jamshidi M. (2001); *Intelligent Control Systems using Soft Computing Methodologies*, CRC Press LLC, ISBN 0-8493-1875-0.