IOP Conf. Series: Journal of Physics: Conf. Series **1487** (2020) 012045 doi:10.1088/1742-6596/1487/1/012045

Robot Positioning Error Compensation Method Based on Deep Neural Network

Junshan Hu*, Fangfang Hua and Wei Tian

College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China.

*Corresponding author email: hujunshan@nuaa.edu.cn

Abstract. Industrial robots are widely used in intelligent manufacturing industry because of their high efficiency and low cost, but the low absolute positioning accuracy limits their application in the field of high-precision manufacturing. To improve the absolute positioning accuracy of robot and solve the traditional complex error modeling problems, a robot positioning error compensation method based on deep neural network is proposed. The Latin hypercube sampling is carried out in Cartesian space, and the influence rule of target attitude on error is obtained. A positioning error prediction model based on genetic particle swarm optimization and deep neural network (GPSO-DNN) is established to realize the prediction and compensation of the positioning errors. The experimental results show that the positioning error compensation method based on GPSO-DNN presents good compensation accuracy. The positioning error is reduced from 1.529mm before compensation to 0.343mm, and the accuracy is increased by 77.57%. This method can effectively compensate the positioning error of the robot and greatly improve the positioning accuracy of the robot.

1. Introduction

With the advancement of the "Made in China 2025" strategy, the intelligent manufacturing industry makes high demands for robot application technology. The application of robots in high-precision manufacturing fields such as aircraft assembly, flexible grinding, and laser cutting becomes increasingly widespread. The high-precision tasks performed by robots rely on their absolute positioning accuracy. Commonly robots have a relatively high repeatability accuracy within ±0.06 mm, but a low absolute precision which is only ±1~2 mm [1]. Therefore, the accurate compensation method to improve the absolute positioning accuracy of the robots is an important basis for promoting the application of robots in the field of intelligent manufacturing.

The robot positioning error compensation method, also known as the robot precision compensation method, refers to the errors generated by a certain means to compensate the original errors of robots. According to the robot control method, the existing methods for improving the absolute positioning accuracy of robots can be classified into online detection feedback compensation method and offline calibration feedforward compensation method. The online detection feedback compensation method usually adds external detection device in the robot system to obtain the feedback information from the robot end or joint in real time, thereby realizing the closed or semi-closed loop control to improve the absolute positioning accuracy to 0.2 mm [2,3]. However, such methods are highly dependent on extra monitoring equipment and are difficult to implement in the complex industrial workspace.

The offline calibration feedforward compensation method is further divided into a kinematic model

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

calibration method and a non-kinetic model calibration method. The basic principle of the kinematic model calibration method [4,5] is to obtain the kinematic parameter errors of the robot through certain measurement methods and parameter identification methods to correct the kinematics model of the robot. The shortcoming of this method is that the modeling and parameter identification process are complicated. This model only considers the geometric error source which only accounts for 80%~90% of the total errors [6]. Thus the precision compensation effect is limited.

Non-kinetic model calibration methods such as neural network method [7], spatial interpolation method [8], etc., not only consider geometric error factors but also include non-geometric error factors such as gear gap, load variation, and thermal effects. Zhou et al. [8] proposed a method of accuracy compensation based on spatial interpolation, which is compensated by spatial interpolation to estimate the position error at the target point of the robot. However, the compensation effect of this method is significantly affected by the size of sampling step. Xu et al. [9] used the feedforward neural network to predict the joint angle error and applied it to the control system for error correction. Nguyen et al. [10] proposed to use artificial neural networks to compensate for non-geometric errors, and three joint angles is employed as neural network inputs to get predicted values of non-geometric errors. Wang et al. [11] established the mapping between the actual coordinates of the target points and the theoretical coordinates through the ELM algorithm, and the absolute positioning accuracy of the robot was improved by 45%. But the spatial sampling range of the method is too small to apply in practical engineering tasks. The above methods do not take the influence of the target point attitude on the positioning error into account, so the compensation accuracy is limited.

Aiming at the above problems, this paper proposes a method based on genetic particle swarm optimization to optimize the depth of neural network (GPSO-DNN) for robot positioning error compensation. Furthermore, the influence of target position and attitude on positioning error is also considered. The target positioning error is predicted and compensated. The proposed GPSO-DNN model is also compared with the models such as genetic algorithm optimization (GA) and particle swarm optimization (PSO) to verify the accuracy and practicality of the error compensation.

2. Spatial sampling and error analysis

2.1. Latin hypercube sampling

Since the sampling point data directly reflects the original state of the robot positioning error, the sampling point planning has a significant impact on compensation effect of the robot positioning error. Therefore, a rational sampling point planning method is one of the key steps to ensure the accuracy of the robot positioning error compensation.

Latin Hypercube Sampling (LHS) [12,13] is a statistical method used in sampling experiments to select a series of test points which are evenly distributed in the sampling space. The basic principle of LHS defines that the parameter dimension is N, and the number of sampling times is M. Firstly, the definition domain of each parameter is divided into M non-overlapped interval, within which a parameter value is randomly selected as a sample. Then M samples $\{s_{11}, s_{12}, \cdots, s_{1M}\}$ of the parameter s_1 is randomly combined with M samples $\{s_{21}, s_{21}, \cdots, s_{2M}\}$ of the parameters s_2 to create M binary-element sets $\{s_{1j}, s_{2j}\}(j = 1,2, \cdots, M)$. Thirdly, the binary-element sets $\{s_{1j}, s_{2j}\}(j = 1,2, \cdots, M)$ is randomly paired with another M samples $\{s_{31}, s_{31}, \cdots, s_{3M}\}$ of the parameters s_3. The above random pairing operation is continued until M N-element sets $\{s_{1j}, s_{2j}, \cdots, s_{Nj}\}(j = 1,2, \cdots, M)$ are obtained. Finally, M Latin hypercube samples are achieved and can be expressed as the following formula:

$$\boldsymbol{S} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1M} \\ s_{21} & s_{22} & \cdots & s_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ s_{N1} & s_{N2} & \cdots & s_{NM} \end{bmatrix} \tag{1}$$

In Eq. (1) each column of the matrix $\boldsymbol{S}$ represents a sample vector obtained from a Latin hypercube sample, and each row represents an arbitrary permutation and combination of M sample values of each parameter.

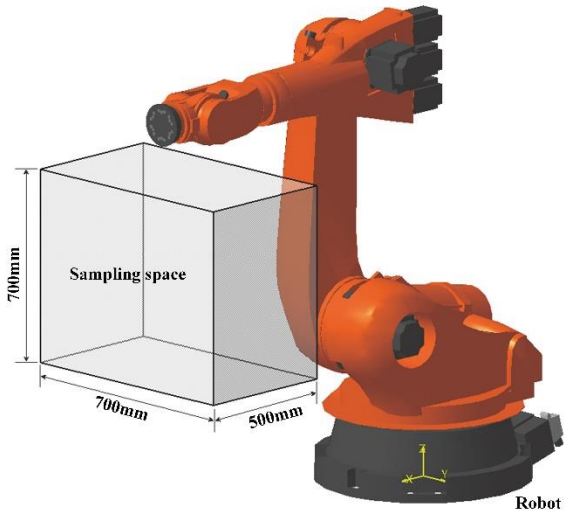
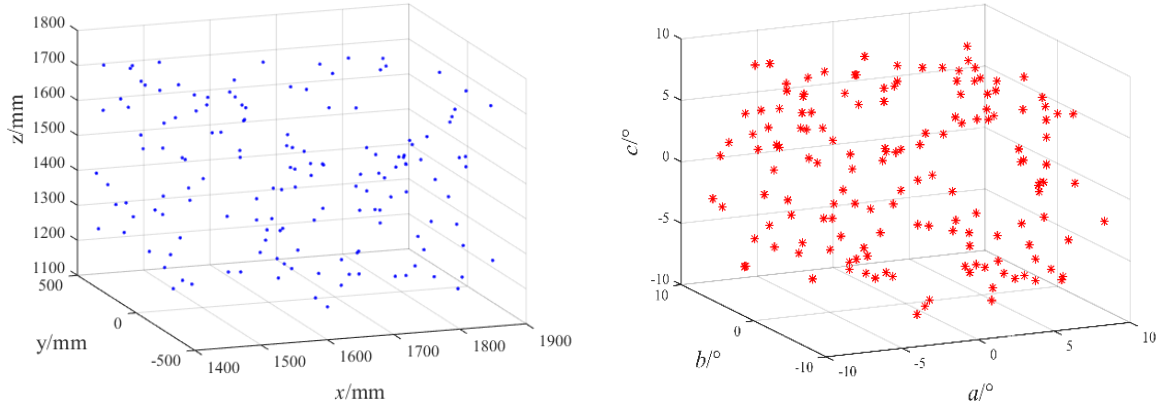


Figure 1. Schematic diagram of the sampling range of the robot Cartesian space.

In order to ensure that the sampling points can reflect the whole working space of robot as much as possible, the LHS method is used to sample in the Cartesian space of the robot as illustrated in Figure 1. A rectangular parallelepiped area with the size of 500 mm × 700 mm × 700 mm is scheduled for as sampling space, in which the posture angles of the potential target points (a, b, c) are all within ±10°. Thus the parameter N equals to 6. When the sampling number M is set to be 150, the sampling points obtained by LHS method are presented in Figure 2. It can be observed that the sampling points obtained by the LHS method are evenly distributed in the sampling space.



(a) Sampling point set in *x*, *y*, *z* direction (b) Sampling point set in *a*, *b*, *c* direction

Figure 2. Schematic diagram of the LCC sampling points in the robot Cartesian space.

2.2. Error analysis

In the robot kinematic space, the theoretical position coordinate of a certain target point is $\boldsymbol{P}_t(x_t, y_t, z_t)$, and the measured actual position coordinate is $\boldsymbol{P}_a(x_a, y_a, z_a)$. Thus the positioning error vector $\boldsymbol{E}$ of the point can be calculated by:

$$\boldsymbol{E} = \boldsymbol{P}_t - \boldsymbol{P}_a = (x_t - x_a, y_t - y_a, z_t - z_a) = (\Delta x, \Delta y, \Delta z) \tag{2}$$

The absolute positioning error of the certain target point is expressed by the Euclidean distance:

$$e = |\boldsymbol{E}| = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \tag{3}$$

The related research [14] reports that for six-degree-of-freedom rotary joint robots, the first three joints mainly affect its positional accuracy, and the latter three joints affect its attitude accuracy.

However, in practical applications, it is found that the latter three joints also have effects on the positioning accuracy of the robot due to error factors such as installation and load. Therefore, it is essential to study the influence of the target point attitude on the positioning accuracy.

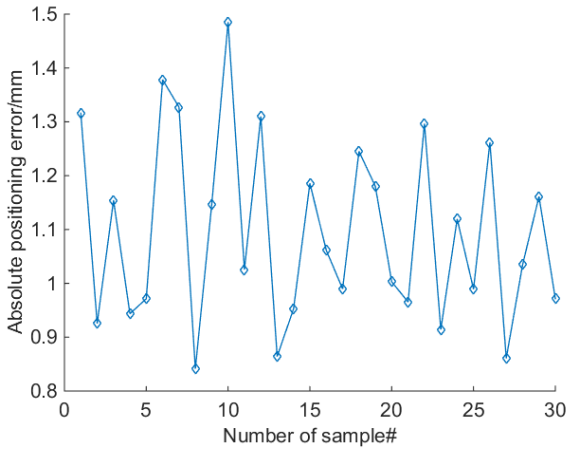


Figure 3. 30 sets of absolute positioning errors at point $(x1962.3, y257.1, z591.4)$.

According to the proposed LHS method, 10 sampling points are planned in the working space of KUKA KR500-3 industrial robot as illustrated in Figure 1. Each sampling point corresponds to 30 sets of attitudes (a, b, c within $\pm 15°$). Thus the absolute positioning errors of 300 target points are measured by the laser tracker. The absolute positioning error under the 30 sets of attitude at the sampling point $(x1962.3, y257.1, z591.4)$ is shown in Figure 3. It can be observed that the absolute positioning error of the target point is distributed in the range of 0.80 mm to 1.50 mm due to the difference of the attitude angle. The error ranges for 30 sets of attitude angles at the above 10 points are lasted in Table 1, in which the maximum of error range is [0.84, 1.49] with a difference of 0.65 mm. It can be found that there is large difference in positioning errors among the distinct positioning angles at the same positioning point, which indicates that that the change of the target point posture has a great influence on the accuracy of the robot. Therefore, the positioning error compensation of the robot must also fully consider the impact of target attitude on positioning accuracy.

Table 1. The absolute positioning error under the 30 sets of attitude at a certain sampling point.

Coordinates of positioning points /mm	Tolerance scope /mm
(1908.11,229.28,1302.13)	[0.42,0.68]
(1996.32, -17.10, 1137.11)	[0.56,0.75]
(2170.25, -1200.97, 744.01)	[0.71,1.21]
(2211.21, -522.68, 1430.81)	[0.19,0.47]
(2234.43, -604.83, 1206.02)	[0.33,0.66]
(2336.65, -162.73, 1038.75)	[0.49,0.78]
(2184.11, -1246.43, 816.61)	[0.62,1.08]
(2290.27, -424.06, 1068.87)	[0.43,0.70]
(1962.33, 257.11, 591.39)	[0.84,1.49]

Therefore, in order to ensure the machining precision, the sampling points should be reasonably planned so that the sampling points can be evenly distributed in the machining space. Besides, the error compensation method should also consider the influence of target position and attitude to ensure the precision compensation effects.

3. Positioning error compensation method based on improved DNN algorithm

3.1. Genetic particle swarm optimization

The particle swarm optimization (PSO) presents good convergence property in the early stage but has a low efficiency in the late evolutionary. Aimed at this shortcoming of PSO, the paper presents a genetic particle swarm optimization(GPSO) algorithm, which introduces crossover factor of the choice of genetic algorithm to improve the convergence speed and global optimization performance.

Particle Swarm Optimization [15] is a global optimization algorithm derived from the simulated predation behaviour of birds. The optimal solution is found in the solution space through sharing individual information and group iteration in the population. The search space dimension is set to be D, and population size is set to be n. The particle position $\boldsymbol{X}_i$ and velocity vector $\boldsymbol{V}_i$ of particle i can be expressed by:

$$\boldsymbol{X}_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^T \ (i = 1,2, \cdots, n) \tag{4}$$

$$\boldsymbol{V}_i = (v_{i1}, v_{i2}, \cdots, v_{iD})^T \ (i = 1,2, \cdots, n) \tag{5}$$

During each iterative evolution, the particles update their speed and position through individual extremum and population extremum. The formula is updated as follows:

$$\boldsymbol{V}_i^{t+1} = \omega \boldsymbol{V}_i^t + c_1 r_1 (\boldsymbol{P}_i^t - \boldsymbol{X}_i^t) + c_2 r_2 (\boldsymbol{P}_g^t - \boldsymbol{X}_i^t) \tag{6}$$

$$\boldsymbol{X}_i^{t+1} = \boldsymbol{X}_i^t + \boldsymbol{V}_i^{t+1} \tag{7}$$

where $\boldsymbol{P}_i$ is the individual extremum, and $\boldsymbol{P}_g$is the group extremum. t is the current iteration number. ω is the inertia weight. c_1 and c_2 are the non-negative constant learning factors. $r_1$and r_2 are the random numbers in the interval [0,1].

The proposed GPSO algorithm draws on the idea of genetic algorithm's choice of crossover, and uses the crossover operation to generate new populations, which expands the search space of particle swarms. Each time when the particle swarm position and velocity is updated, the first half of the population with better fitness is selected as the next generation. Meanwhile, the same part of the better-fitted particles is employed to generate progenies by crossover operation. Then the half of the better-fitted particles in progenies and their parent generation is selected to be the next generation to realize the renewal of the particle swarm. Since the particles are real-numbered, the crossover-operation method is a kind of real-number crossover method. The particles in the population are randomly paired, and the new particles are obtained with a certain probability p. The position and speed are renewed as follows:

$$\boldsymbol{X}_{child_1} = (1-p) \cdot \boldsymbol{X}_{parent_1} + p \cdot \boldsymbol{X}_{parent_2} \tag{8}$$

$$\boldsymbol{X}_{child_2} = p \cdot \boldsymbol{X}_{parent_1} + (1-p) \cdot \boldsymbol{X}_{parent_2} \tag{9}$$

$$\boldsymbol{V}_{child_1} = \left|\boldsymbol{V}_{parent_1}\right| \times \frac{\boldsymbol{V}_{parent_1} + \boldsymbol{V}_{parent_2}}{\left|\boldsymbol{V}_{parent_1} + \boldsymbol{V}_{parent_2}\right|} \tag{10}$$

$$\boldsymbol{V}_{child_2} = \left|\boldsymbol{V}_{parent_2}\right| \times \frac{\boldsymbol{V}_{parent_1} + \boldsymbol{V}_{parent_2}}{\left|\boldsymbol{V}_{parent_1} + \boldsymbol{V}_{parent_2}\right|} \tag{11}$$

In above equations, $\boldsymbol{X}_{child_k}$ and $\boldsymbol{X}_{parent_k}$ are the positions for children and parent particles, separately. $\boldsymbol{V}_{child_k}$ and $\boldsymbol{V}_{parent_k}$ are the speeds for children and parent particles, respectively. the crossover probability p is a random number within [0,1].

3.2. GPSO optimized DNN algorithm

Deep neural network (DNN) is the simplest but most widely used model in the field of deep learning. It is essentially a multi-layer perceptron with multiple hidden layers. Adjacent joints are fully

connected and network parameter adjustment is performed through error backpropagation algorithm. A DNN network structure with 3 hidden layers is shown in Figure 4.

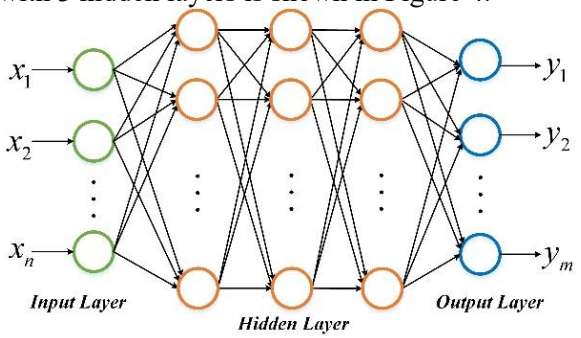


Figure 4. Schematic diagram of the DNN network structure with 3 hidden layers.

The number of nodes in the input and output layers of the DNN is determined by the dimensions of the input vector $[x_1, x_2, \cdots, x_n]$ and the output vector $[y_1, y_2, \cdots, y_m]$, respectively. The number of hidden layers is decided by the number of nodes to be solved and the characteristic of sample data. It is assumed that the DNN contains L hidden layers, thus the the l^{th} $(l = 1, \cdots, N+1)$ output layer is expressed by:

$$A^l = \sigma\left(W^l \bullet A^{l-1} + b^l\right) \tag{12}$$

where W^l is the connection weight matrix between the $(l-1)^{th}$ layer and the l^{th} layer node, and b^l is the threshold vector of the first layer node. The function σ(x) is a modified linear unit (ReLU). It can be expressed in the following formulation:

$$ReLU(x) = \begin{cases} x & if(x \geq 0) \\ 0 & if(x < 0) \end{cases} \tag{13}$$

Compared with the conventional sigmoid or tanh activation function, the Eq. (15) can brings about better convergence and sparsity to the network.

Related studies [16] have shown that the number of hidden layer and nodes of DNN has a significant influence on network performance and prediction results. As the number of hidden layers or nodes increases, the network accuracy is improved. But it takes more training time, and the "over-fitting" phenomenon is prone to occur. The deep network has higher parameter efficiency than the shallow network, and the DNN network has higher redundancy. The hidden layer number reduction strategy is beneficial to improve the efficiency of the network parameters and reduce training time. At present, there is no exact method to determine the optimal number of hidden layers and the number of nodes. Generally, the range of the optimal hidden layer nodes is roughly determined by referring to the following rule of thumb [17]:

$$L = \sqrt{(n+m)} + a \tag{14}$$

where, n and m are the input and output layer nodes, respectively. The constant parameter $a = \{1,2,\cdots,10\}$. The optimal number of hidden layers and the number of nodes are determined by iterative experiments on the elements traversing their value intervals.

The initialization method of weight and threshold is also crucial to the convergence of neural network algorithm. However, the commonly used random initialization method is easy to put the parameter search range of network training in the non-optimal interval, which leads to slow convergence speed. Therefore, this paper proposes an improved deep neural network algorithm (GPSO optimized DNN) to obtain the optimal initial weights and thresholds of DNN, so that the optimized DNN model can better achieve data prediction. In present research, the weight and threshold is depicted by particle position in the particle swarm, and the mean square error between the actual output of the DNN and the expected output as the fitness function of the GPSO algorithm, which is shown in the following formulation:

$$f = \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{y}_i - \boldsymbol{y}_i')^2 \tag{15}$$

where y_i represents the actual value of the training sample and y_i' represents the network output value of the training sample. It is apparent that the smaller the fitness function value, the smaller the prediction error of the DNN model.

The procedure of GPSO optimized DNN algorithm is described as follows:

(1) Normalize sample data and divide them into training samples and testing samples;

(2) Determine the value range of the hidden layers and nodes, and set the initial value of them;

(3) Use the GPSO algorithm to optimize the DNN, and obtain the optimal weights and thresholds under the current network structure;

(4) Conduct network training and testing, and save network structure and prediction results;

(5) Try all the values in the range of the hidden layer and node number. If the result does not satisfy the end condition, return to the procedure 3. Otherwise stop training and select the network structure and prediction data with the highest prediction accuracy.

The algorithm flow of GPSO optimization DNN is shown in Figure 5.

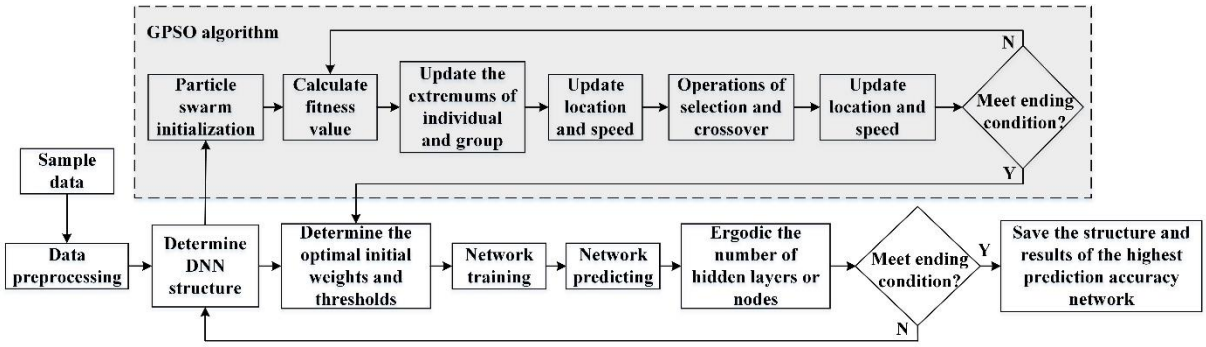


Figure 5. GPSO optimization DNN algorithm flow chart.

3.3. Robot positioning error compensation method based on improved DNN

Different from the traditional kinematic model calibration which only considers geometric error sources, the deep neural network model can simultaneously take both geometric and non-geometric error sources into account. Thus the complex mapping relationship between input and output is well established and robot positioning error is accurately predicted.

In fact, the various error sources of the robot are comprehensively reflected in the positioning error. As aforementioned, the robot positioning error is not only related to the position of the target point but also related to the target point posture. Here the theoretical pose of the target point $\boldsymbol{P}^t = (x^t, y^t, z^t, a^t, b^t, c^t)^T$ is used as the input of the DNN, and the actual positioning error of the target point $\boldsymbol{E} = (\Delta x, \Delta y, \Delta z)^T$ is taken as the output. Therefore, the number of the input layer node is 6, and the number of the output layer node is 3. The optimal hidden layer node range is determined to [4,13] according to Eq. (14). To further ensure network accuracy, the range of hidden layer nodes is appropriately expanded to [3,16], Since the robot accuracy compensation is not extremely complicated, the range of hidden layer layers is set to [1,6].

The robot positioning error compensation method based on improved DNN is a feedforward control compensation method. The principle is shown in Figure 6. Firstly, the sampling points $\boldsymbol{P}_{sample}$ is planned according to the Latin hypercube sampling method in section 2.1, and the actual positioning error of the sampling points $\boldsymbol{E}_{sample}$ is measured by API radian laser tracker. Then the GPSO-DNN model is trained by the theoretical pose and actual positioning error data $\boldsymbol{P}_{sample}$ and $\boldsymbol{E}_{sample}$. Thirdly, the predicted positioning error of the target points $\widehat{\boldsymbol{E}}_{target}$ is obtained by putting the theoretical coordinate of the target points into the well-trained model. Finally, the predicted value of

the positioning error is superposed on the theoretical coordinates of the target points, and the modified coordinates of target points are transmitted to the robot for compensation.

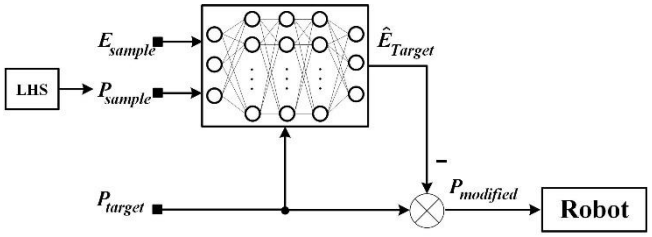


Figure 6. Schematic diagram of robot positioning error compensation based on improved DNN.

4. Experimental verification and analysis

4.1. Coordinate system establishment and data acquisition

The test platform used to test and verify the robot positioning error compensation method based on the improved DNN algorithm is built as shown in Figure 7. KUKA kr500-3 robot is used as carrier and API radian laser tracker is employed as measuring device. The laser tracker target ball is placed in a fixed position of the end effector carried by the robot. The repeated positioning precision of the robot is ±0.06mm, and the absolute positioning precision of laser tracker is ±0.06 μm+3.5μm/m.

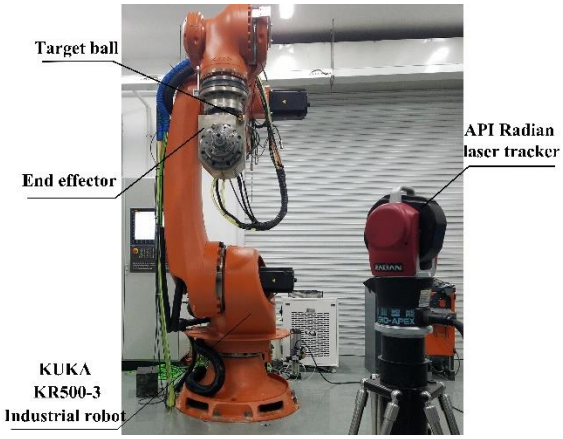


Figure 7. Test Platform.

The theoretical pose of the target point is essentially the coordinate conversion relationship between the tool coordinate system and the robot base coordinate system. The robot base coordinate system, the flange coordinate system and the tool coordinate system are established by API laser tracker. The establishment procedure of robot base coordinate system is as follows:

(1) Rotate the A1 axis while keep the other axes stationary. Use the laser tracker to obtain the position data of the target ball point during rotation process, and fit them into the circle O_1. Use the same method to rotate the A2 axis only and obtain the circle O_2.

(2) Make a plane 1 parallel to the circle O_2 through the center of the circle O_1. Project the center of the circle O_2 on plane 1 to obtain the projection point ${C_2}'$.

(3) Make a plane 2 parallel to the circle O_1 through the projection point ${C_2}'$. Translate plane 2 down 1045 mm along its normal direction to create the robot base plane. The normal direction of this plane is the Z axis of the base coordinate system.

(4) Project the center of circle O_1 and the point ${C_2}'$ onto the robot base plane to obtain the origin of the base coordinate system and a point on the X axis, respectively.

(5) The base coordinate system of the robot is established by the origin, the point on the X axis and the Z axis.

A testing space with a size of 600 mm × 1200 mm × 800 mm and is planed within the range of robot motion. 2000 target points and 100 verification points are randomly generated according to the Latin hypercube sampling method. The target point attitude angle (a, b, c) in this testing space is within the range of ±10°. The actual positioning errors of the above 2000 target points are obtained by using a laser tracker for model verification.

4.2. Test verification and model comparison

The theoretical postures of the target points are taken as the input of the DNN, and the actual positioning errors of the target points are used as the output of the DNN to train the proposed model. 1900 target points are used as training samples and the other 100 target points are used as verification samples. The GA-DNN and PSO-DNN models are compared with the proposed GPSO-DNN model. The maximum training number of the model is 500. The learning rate is 0.01, and the minimum error of the training target is 10^{-5}.

The network parameters of the above three models were determined by trial and error. In the proposed model, the number of hidden nodes in the proposed model is [20, 10, 5]; the population size is 20; the evolutionary algebra is 50; the learning factors c_1 and c_2 are 1.4962; both the individual velocity and position range are [1,1], the crossover factor is 0.2. In GA-DNN model, the population size 50; the evolutionary algebra is 100; the crossover factor p is 0.3, the mutation probability is 0.1. In PSO-DNN model, the population size is 40; the evolutionary algebra is 100; the learning factors c_1 and c_2 are 1.4962; both the individual speed and location range are [1,1].

After the definition of network parameters and model training, the predicted position errors obtained by three model are added to the theoretical position coordinates of the verification samples to achieve the modified position coordinates. These modified positioning points are converted to the robot controller to guide robot's movements. Meanwhile, the laser tracker is employed to capture the real position of the robot. The positioning errors are calculated according to Eq. (3) and presented in Figure 8. The error compensation of 100 verification points obtained from the three models are compared in Table 2.

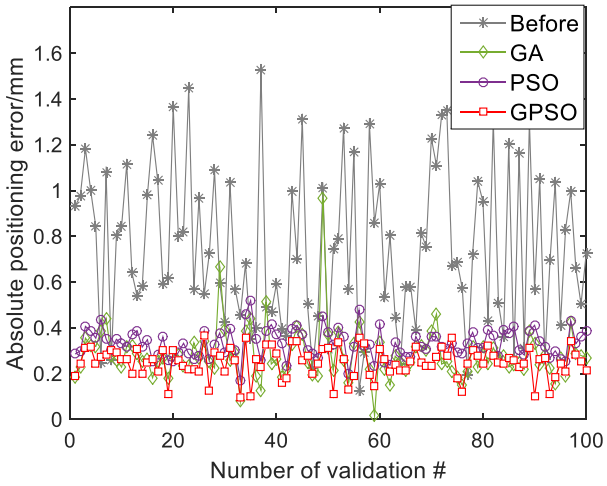


Figure 8. Comparison of absolute positioning errors before and after model compensation.

Table 2. Comparison of data compensation results of three models

Error（mm）	Max	Min	Average	Standard deviation
Uncompensated	1.529	0.124	0.754	0.340
GA-DNN	0.965	0.017	0.284	0.114
PSO-DNN	0.519	0.172	0.333	0.062
GPSO-DNN	0.364	0.097	0.249	0.064

It can be seen that the absolute positioning errors of the verification points are significantly smaller than those before compensation, and the compensation accuracy of GPSO-DNN is higher than the other two models. After the compensation of the GPSO-DNN model, the maximum absolute positioning errors of the 100 verification points are reduced from 1.529 mm to 0.364 mm, and the average value is reduced from 0.754 mm to 0.249 mm. The maximum absolute positioning error of the robot is reduced by 76.19%, and the standard deviation is only 0.034 mm.

5. Conclusion

In present research, the Latin hypercube sampling method is employed to conduct sampling plan of the robot workspace. A robot positioning error compensation method is proposed based on the improved deep neural network (GPSO-DNN). The experimental verification results show that the GPSO-DNN based positioning error compensation method has good compensation accuracy. The positioning error is reduced from 1.529 mm to 0.364mm, and the positioning accuracy of the robot is greatly improved by 76.19%, which verifies the practicality and accuracy of the method.

Reference

[1] Olsson T, Haage M, Kihlman H, Johansson R, Nilsson K, Robertsson A, et al. Cost-efficient drilling using industrial robots with high-bandwidth force feedback. Robot Cim-Int Manuf. 2010; 26(1):24-38.

[2] Saund B, Devlieg R. High Accuracy Articulated Robots with CNC Control Systems. Sae International Journal of Aerospace, 2013, 6(2):780-784.

[3] Kihlman Henrik, Loser Raimund, Cooke Andrew, et al. Metrology-integrated industrial robots: calibration, implementation and testing, Proceedings of the 35th ISR (International Symposium on Robotics), 2004.

[4] Zak G, Benhabib B, Fenton R G, et al. Application of the weighted least squares parameter estimation method to the robot calibration. Journal of Mechanical Design, 1994, 116(3):890-893.

[5] Roth Z S, Mooring B W, Ravani B. An Overview of Robot Calibration. Information Technology Journal, 1987, 3(1):377-385.

[6] Renders J M , Rossignol E , Becquet M , et al. Kinematic calibration and geometrical parameter identification for robots. Robotics & Automation IEEE Transactions on, 1991, 7(6):721-732.

[7] Zhong X, John Lewis, Francis L. N-Nagy. Inverse robot calibration using artificial neural networks. Engineering Applications of Artificial Intelligence, 1996, 9(1):83-93.

[8] Wei Z, Wenhe L, Wei T. Theory and experiment of industrial robot accuracy compensation method based on spatial interpolation. Journal of mechanical engineering. 2013,49(03):42-48.

[9] Xu W L, Wurst K H, Watanabe T, et al. Calibrating a modular robotic joint using neural network approach. IEEE World Congress on IEEE International Conference on Neural Networks. 1994.

[10] H.N. Nguyen, J. Zhou, H.J. Kang, A calibration method for enhancing robot accuracy through integration of an extended Kalman filter algorithm and an artificial neural network, Neurocomputing 151 (2015) 996–1005.

[11] Zhang LF, Li X, Zhang LY, Ye N. Analysis of the Positioning Error of Industrial Robots and Accuracy Compensation Based on ELM Algorithm. Robot. 2018, 40(06):77-85+93.

[12] Dam E R V , Husslage B , Melissen H H . Maximin Latin Hypercube Designs in Two Dimensions[J]. Operations Research, 2007, 55(1):158-169.

[13] Mckay M D , Conover R J B J . A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code[J]. Technometrics, 1979, 21(2):239-245.

[14] Baillieul J. Introduction to ROBOTICS mechanics and control. IEEE Transactions on Automatic Control, 1987, 32(5):463-464.

[15] Kennedy J, Eberhart R. Particle swarm optimization. IEEE International Conference on Neural Networks, 1995,(4):1942~1948.

[16] Zhang S , Liu C , Jiang H , et al. Nonrecurrent Neural Structure for Long-Term Dependence. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2017, 25(4):871-884.

[17] Jeong R, Rilett L R. Bus arrival time prediction using artificial neural network model. International IEEE Conference on Intelligent Transportation Systems. 2004.