Proceedings of the 4th WRC Symposium
on Advanced Robotics and Automation 2022
Beijing, China, September 20, 2022

2022 WRC Symposium on Advanced Robotics and Automation (WRC SARA) | 978-1-6654-6369-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/WRCSARA57040.2022.9903922

# Calibration of Serial Robots through Integration of Local POE Formula and Artificial Neural Networks

Yongbin Song, Yanling Tian, and Yiwei Ma

*Abstract*— **This paper presents a calibration method for serial robots to improve pose accuracy. In this method, the complicated calibration of the actual robot containing various error sources is converted to a simple one of the equivalent robot only containing configuration-dependent joint motion errors. For the lower-mobility robot, which has *n* (less than 6) degree of freedom, other 6-*n* virtual joints need to be introduced into the equivalent robot to meet the completeness requirement. A simplified local POE formula is used to build the relationship between the pose error and joint motion errors, and an artificial neural network is used to approximate the relationship between joint motion errors and nominal joint variables. By integrating the two models, the error model of the equivalent robot can be deduced and then used for calibration. Simulation results on a typical serial robot show that the proposed method can reduce pose errors significantly.**

## I. INTRODUCTION

In recent years, robotic manipulators have played an important role in machining large complex components in several growing industrial sectors, e.g., aeronautics, astronautics, and railroad [1]. This is because industrial robots have the advantages of both large operation reach and low manufacturing costs compared to conventional machine tools. However, the absolute pose accuracy, which is one of the most important performance specifications of robots, remains to be improved for high-precision processing [2].

It has been observed that if sufficient repeatability can be guaranteed via the manufacturing and assembly process, calibration is a highly cost-efficient technique to improve pose accuracy through software [3]. The main procedures can be summarized as follows. Firstly, a mathematical model that consists of a sequence of adjustable parameters needs to be established to predict the actual pose or pose error. Then, these parameters can be determined with the measurement information of finite configurations. Finally, with the identified model at hand, it is necessary to modify the nominal inverse kinematics or construct a compensator in order to compensate for pose errors at any possible configurations. Over the past few decades, there has been a great deal of intensive literature on the calibration of robotic manipulators. The available methods can be classified into two categories: model-based and data-driven methods.

The model-based method generally involves building the relationship between the pose error and error sources by differentiating the forward kinematics with respect to kinematic parameters. By minimizing pose errors at certain configurations, error sources can be identified and an approximation of the actual kinematics can be found. There are two basic approaches to formulate forward kinematics. The first approach uses the multiplication of a set of homogeneous transformation matrices based on D-H convention [4]. To solve parametric discontinuities arising from nearly parallel neighboring joint axes in the D-H model, various modified methods have been proposed [5,6]. The second approach uses a product of matrix exponentials (POE) by drawing upon screw theory [7,8]. Kinematic parameters of the POE model vary smoothly and can be defined either in the global sense or in the local sense. Although these model-based methods have significant physical meanings, they still have certain limitations. For instance, they have to take both encoder offsets and structural errors into consideration under the requirements of completeness, resulting in a complicated modeling process. In addition, error parameters are usually redundant and need to be minimized either at the model level or at the identification level [9,10]. Moreover, error parameters are actually not constant because they arise from not only geometric errors but also non-geometric errors such as pitch errors and elastic deformations due to gravity [11].

In contrast, the data-driven method does not need to model error sources but simply treats the pose error as a function with configuration. It requires the function to be approximated by using spatial data interpolation/fitting approaches. The interpolation methods include inverse distance weighted method [12], kriging method [13], and fuzzy method [14]. Specifically, the task space can be divided into several cubic cells, and then pose errors on the grid points in each cell are measured and stored in memory, for interpolation and compensation of pose errors at any possible configurations. Some fitting methods such as Fourier polynomial [15] and artificial neural network [16,17] have also become available. However, the calibration accuracy is usually positively related to the number of measuring configurations since no prior knowledge of the robot is put to use. Thus, it would require a huge size of memory space and quite a time-consuming measurement process.

In order to combine the virtues of both model-based methods and data-driven methods, we propose the idea of building an equivalent robot for calibration. Specifically, the pose error arising from various error sources in the actual robot is treated as that generated only by joint motion errors in the equivalent robot. Here, the joint motion errors are defined as the errors of the nominal joint variables in the equivalent robot, but they do not have realistic physical meanings in the actual robot. It is worth noting that this idea can simplify the error modeling process due to the neglect of structural error sources. However, the cost of this simplification is that joint motion errors can no longer be viewed as constants, but can be seen as

This work is supported by EU H2020-RISE-ECSASDP (grant 734272).
(Corresponding author: Yiwei Ma.)
The authors are with the school of engineering, University of Warwick, Coventry, CV47AL, UK (e-mail: yongbin.song@warwick.ac.uk; y.tian.1@warwick.ac.uk; yi-wei.ma@warwick.ac.uk).

configuration-dependent variables. Therefore, it requires the relationship between joint motion errors and nominal joint variables to be further approximated by a data-driven method. From the view of compensation, when the joint motion errors are compensated, the pose output of the actual robot should be identical to the desired pose output.

The rest of this paper is organized as follows. In Section II, we briefly review the modeling of the actual robot by using local POE formula and also point out limitations. In Section III, the equivalent robot is built by integrating local POE formula and artificial neural networks. In Section IV, the calibration procedures are described based on the equivalent error model. Section V carries out the simulation on a 6-DOF serial robot to verify the effectiveness of the proposed method, before conclusions are drawn in Section VI.

## II. Modeling of the Actual Robot

In this section, the local POE-based method which is trying to model error sources of the actual robot is briefly reviewed. Firstly, the forward kinematic model of serial robots is established. Then, the linearized error model is established by differentiating the forward kinematic model. Finally, the limitations are pointed out.

### A. Forward Kinematic Model based on Local POE Formula

Fig. 1 shows an $n$-degree-of-freedom (DOF) serial chain containing either revolute or prismatic joints. Let link $j-1$ and link $j$ be adjacent and connected by joint $j$. A world frame $\mathcal{K}_0$ and a tool frame $\mathcal{K}_{n+1}$ are located at the base and the end effector, respectively. In order to describe the error sources in local frames, the body-fixed frame $\mathcal{K}_j$ is attached to link $j$ with its $z$ axis coinciding with the axis of joint $j$.

At a given configuration, the pose (position and orientation) of $\mathcal{K}_{n+1}$ evaluated in $\mathcal{K}_0$, which is an element of $SE(3)$, can be expressed by multiplications of a sequence of $4\times4$ homogeneous transformation matrices, that is

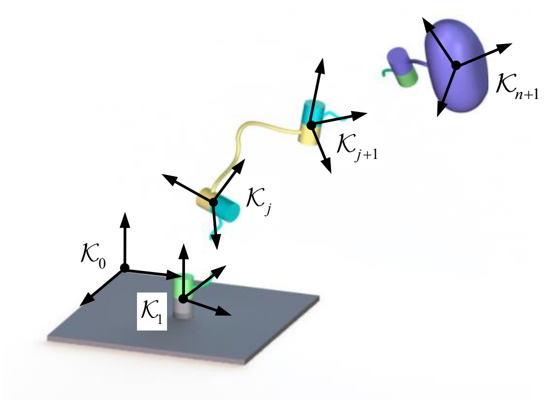


Figure 1. Schematic diagram of an $n$-DOF serial chain.

$$ {}^{0}\boldsymbol{A}_{n+1} = \prod_{j=1}^{n+1} {}^{j-1}\boldsymbol{A}_j \tag{1}$$

where ${}^{j-1}\boldsymbol{A}_j$ represents the pose of $\mathcal{K}_j$ relative to $\mathcal{K}_{j-1}$. Note that ${}^{j-1}\boldsymbol{A}_j$ can be obtained by applying the screw motion of joint $j$ on the initial relative pose ${}^{j-1}\boldsymbol{A}_j(0)$, which is specified when the $j$th joint variable is set to zero. Then, ${}^{j-1}\boldsymbol{A}_j$ can be further expressed as

$$ {}^{j-1}\boldsymbol{A}_j = {}^{j-1}\boldsymbol{A}_j(0)\, e^{\hat{\xi}_{j,j} q_j}, \quad j = 1 \sim n \tag{2}$$

where $q_j$ is the $j$th nominal joint variable, $\hat{\xi}_{j,j}$ is the $j$th unit joint twist evaluated in $\mathcal{K}_j$. The matrix exponential can map the twist belonging to $se(3)$ to the associated screw belonging to $SE(3)$ as

$$ e^{\hat{\xi}_{j,j} q_j} = \boldsymbol{I}_4 + \hat{\xi}_{j,j} q_j + \left(1-\cos q_j\right)\hat{\xi}_{j,j}^2 + \left(q_j - \sin q_j\right)\hat{\xi}_{j,j}^3 \tag{3}$$

where $\boldsymbol{I}_4$ denotes the fourth-order identity matrix. Moreover, the twist can be expressed either in a matrix form or in a vector form. The isomorphic mapping between them can be described as

$$ \hat{\xi} = \begin{bmatrix} \hat{\boldsymbol{\omega}} & \boldsymbol{v} \\ \boldsymbol{0}^{\mathrm{T}} & 0 \end{bmatrix} \in se(3) \mapsto \xi = \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{pmatrix} \in \mathbb{R}^6 \tag{4}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ and $\boldsymbol{v} \in \mathbb{R}^3$ are two vectors completely defining the joint twist, and $\hat{\boldsymbol{\omega}}$ denotes the skew-symmetric matrix of $\boldsymbol{\omega}$. Since the unit joint twist is evaluated in the local frame, it can be given by

$$ \hat{\xi}_{j,j} = \begin{cases} (0\ 0\ 0\ 0\ 0\ 1)^{\mathrm{T}} & \text{if joint } j \text{ is a revolute joint} \\ (0\ 0\ 1\ 0\ 0\ 0)^{\mathrm{T}} & \text{if joint } j \text{ is a prismatic joint} \end{cases} \tag{5}$$

According to Chasles's theorem, any rigid-body motion can be described as a screw motion associated with a certain twist. This means that we can always find a twist $\hat{\zeta}_{j,j}$ to satisfy $e^{\hat{\zeta}_{j,j}} = {}^{j-1}\boldsymbol{A}_j(0)$. Thus, (2) can be rewritten as

$$ {}^{j-1}\boldsymbol{A}_j = e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j}, \quad j = 1 \sim n \tag{6}$$

Substituting (6) into (1) leads to

$$ {}^{0}\boldsymbol{A}_{n+1} = \left( \prod_{j=1}^{n} e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j} \right) e^{\hat{\zeta}_{n+1,n+1}} \tag{7}$$

Equation (7) establishes the forward kinematic model of the actual serial robot. Examining (7) indicates that the possible error sources in the actual robot can be classified into three categories:

(1) the errors of the initial relative poses $\hat{\zeta}_{j,j}$, i.e., manufacturing tolerances of links and assembly errors;

(2) the errors of the unit joint twists $\hat{\xi}_{j,j}$, i.e., axis misalignments of revolute joints and straightness of prismatic joints;
(3) the errors of the joint variables $q_j$, i.e., encoder offsets and pitch errors.

### B. Linearized Error Model based on Local POE Formula

By differentiating (7) with respect to $\hat{\zeta}_{j,j}$, $\hat{\xi}_{j,j}$, and $q_j$, we have the linearized error model of the serial chain

$$
\begin{aligned}
&\delta\, {}^{0}\boldsymbol{A}_{n+1}\, {}^{0}\boldsymbol{A}_{n+1}^{-1} \\
&= \sum_{j=1}^{n+1} e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} \delta\hat{\boldsymbol{\zeta}}_{j,j} \left( e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} \right)^{-1} \\
&+ \sum_{j=1}^{n} e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j} \delta\hat{\boldsymbol{\xi}}_{j,j} q_j \left( e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j} \right)^{-1} \\
&+ \sum_{j=1}^{n} e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j} \hat{\boldsymbol{\xi}}_{j,j} \delta q_j \left( e^{\hat{\zeta}_{1,1}} \cdots e^{\hat{\zeta}_{j,j}} e^{\hat{\xi}_{j,j} q_j} \right)^{-1}
\end{aligned} \tag{8}
$$

By using the adjoint representation, a more compact form of (8) can be expressed as

$$
\begin{aligned}
&\delta\, {}^{0}\boldsymbol{A}_{n+1}\, {}^{0}\boldsymbol{A}_{n+1}^{-1} \\
&= \sum_{j=1}^{n+1} \mathrm{Ad}_{j0} \delta\hat{\boldsymbol{\zeta}}_{j,j} + \sum_{j=1}^{n} \mathrm{Ad}_{j} q_j \delta\hat{\boldsymbol{\xi}}_{j,j} + \sum_{j=1}^{n} \mathrm{Ad}_{j} \hat{\boldsymbol{\xi}}_{j,j} \delta q_j
\end{aligned} \tag{9}
$$

Drawing upon the isomorphic mapping of (4), (9) can be rewritten as

$$
\delta\boldsymbol{y} = \sum_{j=1}^{n+1} \mathrm{Ad}_{j0} \delta\boldsymbol{\zeta}_{j,j} + \sum_{j=1}^{n} \mathrm{Ad}_{j} q_j \delta\boldsymbol{\xi}_{j,j} + \sum_{j=1}^{n} \mathrm{Ad}_{j} \boldsymbol{\xi}_{j,j} \delta q_j \tag{10}
$$

where $\delta\boldsymbol{y} \in \mathbb{R}^6$ is the pose error vector evaluated in $\mathcal{K}_0$, $\mathrm{Ad}_{j0} \in \mathbb{R}^{6\times 6}$ ( $\mathrm{Ad}_{j} \in \mathbb{R}^{6\times 6}$ ) is the adjoint transformation which maps the twist from initial(current) $\mathcal{K}_j$ to $\mathcal{K}_0$.

According to (10), $\delta\boldsymbol{\zeta}_{j,j} \in \mathbb{R}^6$, $\delta\boldsymbol{\xi}_{j,j} \in \mathbb{R}^6$, and $\delta q_j \in \mathbb{R}$ are the overall kinematic errors in the actual serial robot. Hence, $13n+6$ error parameters need to be identified in total. However, not all of them are identifiable due to redundancy. Therefore, the errors of the unit joint twists and joint variables are neglected for model reduction in [8]. By doing so, (10) can be simplified as

$$
\delta\boldsymbol{y} = \sum_{j=1}^{n+1} \mathrm{Ad}_{j0} \delta\boldsymbol{\zeta}_{j,j} \tag{11}
$$

Although the number of error parameters is reduced to $6n+6$, it has been proved that the model is still not minimal. Some robust estimation methods have to be adopted to address the rank deficiency of the identification matrix. In addition, several studies have investigated how to eliminate the redundant parameters to guarantee the minimality, but the elimination process is quite complicated and requires deep knowledge of mathematics. Another limitation is that the constant error parameters are not able to fully describe all error sources, especially those varying with configurations. Therefore, this kind of purely model-based method may result in relatively low calibration accuracy.

## III. Modeling of the Equivalent Robot

In this section, calibration based on the equivalent robot is described. We assume that the pose errors of the equivalent robot are generated by configuration-dependent joint motion errors. The local POE-based error model is simplified by only keeping joint motion errors. The relationship between joint motion errors and nominal joint variables is approximated by an artificial neural network.

### A. Simplification of Local POE-based Error Model by Keeping Joint Motion Errors

Before we build the simplified error model of the equivalent robot, an important issue of the model completeness needs to be concerned carefully. It is well accepted that the pose error for a given configuration can be expressed by a six-dimensional vector in the task space, which also means it requires at least six parameters to be completely defined. However, there are only *n* parameters of joint motion errors for an *n*-DOF serial robot at a given configuration. Therefore, $6-n$ parameters need to be introduced for a lower-mobility serial robot. They can be treated as joint motion errors of $6-n$ virtual joints in the equivalent robot. Here, the virtual joint is defined as the joint which can generate the theoretically restricted motion of the ideal robot, and its nominal value is zero. It is worth noting that the screw theory can be used to construct virtual joints conveniently. For more details, please refer to [18].

To distinguish between the equivalent robot and the actual robot, we express the errors of the joint variables in the actual robot as joint motion errors $\delta\rho_j$. Then, the simplified error model of the equivalent robot can be given by

$$
\delta\boldsymbol{y} = \sum_{j=1}^{n} \xi_{a,j} \delta\rho_{a,j} + \sum_{j=1}^{6-n} \xi_{c,j} \delta\rho_{c,j} \tag{12}
$$

where $\xi_{a,j}$ ( $\xi_{c,j}$ ) represents the nominal joint twist of the *j*th actuated(virtual) joint evaluated in $\mathcal{K}_0$, $\delta\rho_{a,j}$ ( $\delta\rho_{c,j}$ ) represents the joint motion error of the *j*th actuated(virtual) joint.

Rewriting (10) and (12) into matrix form leads to

$$
\delta\boldsymbol{y} = \boldsymbol{J}_{\zeta}\delta\boldsymbol{\zeta} + \boldsymbol{J}_{\xi}\delta\boldsymbol{\xi} + \boldsymbol{J}_{\boldsymbol{q}}\delta\boldsymbol{q} \tag{13}
$$

$$
\delta\boldsymbol{y} = \boldsymbol{J}_{\boldsymbol{\rho}}\delta\boldsymbol{\rho} \tag{14}
$$

where $\boldsymbol{J}_{\zeta} \in \mathbb{R}^{6\times 6n}$, $\boldsymbol{J}_{\xi} \in \mathbb{R}^{6\times 6n}$, $\boldsymbol{J}_{\boldsymbol{q}} \in \mathbb{R}^{6\times n}$ and $\boldsymbol{J}_{\boldsymbol{\rho}} \in \mathbb{R}^{6\times 6}$ denote the Jacobian matrices from $\delta\boldsymbol{\zeta}$, $\delta\boldsymbol{\xi}$, $\delta\boldsymbol{q}$ and $\delta\boldsymbol{\rho}$ to $\delta\boldsymbol{y}$, respectively.

By comparing (13) and (14), it is easy to have

$$
\delta\boldsymbol{\rho} = \boldsymbol{J}_{\delta\rho}^{-1}\left( \boldsymbol{J}_{\delta\zeta}\delta\boldsymbol{\zeta} + \boldsymbol{J}_{\delta\xi}\delta\boldsymbol{\xi} + \boldsymbol{J}_{\delta q}\delta\boldsymbol{q} \right) \tag{15}
$$

Equation (15) reveals the connection between the actual robot and the equivalent robot. It shows that joint motion errors of the equivalent robot are complicated functions of nominal joint variables and a sequence of constants. These constants include known nominal dimensional parameters and unknown kinematic errors. Instead of identifying the kinematic errors, we treat the relationship between joint motion errors and nominal joint variables as a black box and approximate it by data-driven methods.

### *B. Approximation of Configuration-dependent Joint Motion Errors Using Artificial Neural Networks*

Before we choose the suitable data-driven method, it should be recalled that (15) means that joint motion errors are continuous with nominal joint variables. This is because those Jacobian matrices will always change continuously as long as the robot works on the non-singular path. It is widely known that multilayered feedforward neural networks are regarded as universal approximators that can approximate any arbitrary function to any degree of accuracy, and only one hidden layer is sufficient to approximate continuous functions [19,20]. Therefore, it is reasonable to adopt three-layered feedforward neural networks herein.

As shown in Fig. 2, a three-layered network involves an input layer, a hidden layer and an output layer. The nodes located in the hidden layer, which perform massive nonlinear computations via activation functions, are connected with those located in the input/output layer through adjustable connection weights. Let $n$ elements of nominal joint variables be the inputs, and 6 elements of joint motion errors be the outputs. As a rule of thumb, the number of hidden nodes can be determined as $\sqrt{6n}$. The activation function of the hidden layer is the tan-sigmoid function, whereas that of the output layer is the simple linear function. Then, the proposed neural network can be expressed in an implicit form as

$$\delta\boldsymbol{\rho} = \boldsymbol{N}(\boldsymbol{q}, \boldsymbol{w}) \tag{16}$$

where $\delta\boldsymbol{\rho} \in \mathbb{R}^6$ and $\boldsymbol{q} \in \mathbb{R}^n$ are the vectors of joint motion errors and nominal actuated joint variables, respectively; $\boldsymbol{w} \in \mathbb{R}^N$ is the vector containing all connection weights.

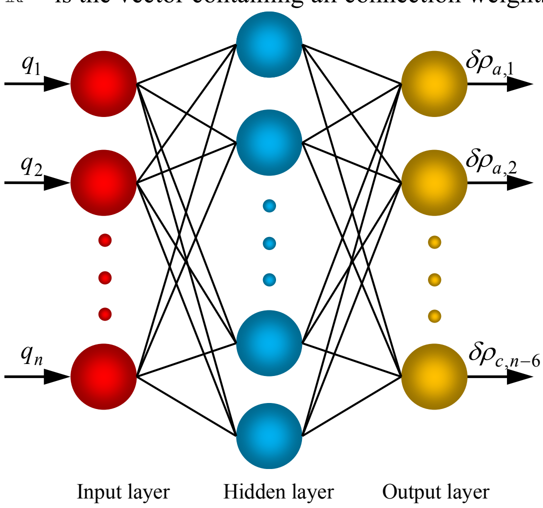


Figure 2. The architecture of the feedforward neural network.

Substituting (16) into (14) leads to

$$\delta\boldsymbol{y} = \boldsymbol{J}_\rho \boldsymbol{N}(\boldsymbol{q}, \boldsymbol{w}) \tag{17}$$

So far, we have established the integrated error model of the equivalent robot. Compared with the traditional local POE-based error model expressed by (13), it has several benefits as follows:

(1) the error modeling process is simplified since it is unnecessary to deduce the Jacobian matrices with respect to joint twists and relative initial pose, and the error parameter elimination is also needless due to the independence of joint motion errors;

(2) the accuracy of the error model is higher because the configuration-dependent joint motion errors are approximated by a strong artificial neural network, which doesn't have an upper bound of fitting accuracy as long as the dataset is enough;

(3) the error compensation is much easier since it only requires correcting the nominal joint variables rather than modifying each local relative frame.

## IV. Calibration Using the Equivalent Robot

In this section, we investigate how to implement calibration using the model of the equivalent robot. Three crucial issues need to be concerned: acquirement of measurement information, training of the artificial neural network, and online compensation of pose errors.

### *A. Pose Measurement*

According to (17), it is necessary to measure the pose error $\delta\boldsymbol{y}$ in several configurations in order to acquire the sample dataset for neural network training. However, to the best knowledge of the authors, there is no existing measuring equipment that can obtain the six-dimensional pose error directly. Therefore, position errors of at least three noncolinear target points located on the end effector are measured to fully determine the pose error. The relationship between them can be expressed as

$$\delta\boldsymbol{p} = \boldsymbol{P}\delta\boldsymbol{y} \tag{18}$$

with

$$\delta\boldsymbol{p} = \begin{pmatrix} \delta\boldsymbol{p}_1 \\ \delta\boldsymbol{p}_2 \\ \delta\boldsymbol{p}_3 \end{pmatrix}, \quad \boldsymbol{P} = \begin{pmatrix} \boldsymbol{I}_3 & \hat{\boldsymbol{r}}_1 \\ \boldsymbol{I}_3 & \hat{\boldsymbol{r}}_2 \\ \boldsymbol{I}_3 & \hat{\boldsymbol{r}}_3 \end{pmatrix}$$

where $\delta\boldsymbol{p}_i \in \mathbb{R}^3$ denotes the position error of the $i$th target point, $\boldsymbol{I}_3 \in \mathbb{R}^{3\times3}$ denotes a third-order identity matrix, $\boldsymbol{r}_i \in \mathbb{R}^3$ denotes the nominal vector pointing from the $i$th target point to the origin of $\mathcal{K}_0$, and $\hat{\boldsymbol{r}}_i \in \mathbb{R}^{3\times3}$ denotes the skew-symmetric matrix of $\boldsymbol{r}_i$.

Substituting (17) into (18) leads to

$$\delta\boldsymbol{p} = \boldsymbol{P}\boldsymbol{J}_\rho \boldsymbol{N}(\boldsymbol{q}, \boldsymbol{w}) \tag{19}$$

### *B. Neural Network Training*

By collecting position errors of three target points in $K$ configurations and recording corresponding nominal joint variables, we have $K$ data pairs to train the neural network. The purpose is to search for the optimal weight vector $\boldsymbol{w}$ which can minimize the following loss function

$$E(\boldsymbol{w})=\frac{1}{K}\sum_{k=1}^{K}\left\|\delta\boldsymbol{p}_k-\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}\boldsymbol{N}(\boldsymbol{q}_k,\boldsymbol{w})\right\|^2 \tag{20}$$

It can be seen that it is similar to solve a nonlinear least squares problem, and thus some nonlinear iterative regression methods may be useful. However, the loss function is formulated in the task space, resulting in repeated calculation of $\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}$ in each iteration. A trick which can avoid this problem is to reformulate the loss function in the joint space as

$$E(\boldsymbol{w})=\frac{1}{K}\sum_{k=1}^{K}\left\|\left(\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}\right)^{+}\delta\boldsymbol{p}_k-\boldsymbol{N}(\boldsymbol{q}_k,\boldsymbol{w})\right\|^2 \tag{21}$$

where $\left(\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}\right)^{+}$ is the pseudo-inverse of $\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}$. By doing this, $\left(\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}\right)^{+}\delta\boldsymbol{p}_k$ is treated as the observations of the joint motion errors $\delta\boldsymbol{\rho}_k$, and thus it only requires one time calculation of $\boldsymbol{P}_k\boldsymbol{J}_{\delta\boldsymbol{\rho},k}$.

Initializing the weight vector $\boldsymbol{w}^{(0)}$ with a normally distributed random vector, it can then be updated by gradient descent method as

$$\boldsymbol{w}^{(l+1)}=\boldsymbol{w}^{(l)}-\alpha\frac{\partial E(\boldsymbol{w})}{\partial\boldsymbol{w}}\bigg|_{\boldsymbol{w}^{(l)}} \tag{22}$$

where $\alpha$ is the learning rate, and the subscript $l$ is the iteration number. The above training process will not terminate until the rate of change of the loss function is below a specified threshold.

### C. Error Compensation

With the trained neural network in hand, online error compensation can be achieved by designing a joint error compensator. The function of the neural network should be written in the kinematic program, and the trained weight vector $\hat{\boldsymbol{w}}$ should be stored in the memory. Given the command pose, the nominal joint variables $\boldsymbol{q}$ can be calculated with nominal inverse kinematics. Then, the joint motion errors can be predicted by

$$\delta\boldsymbol{\rho}=\boldsymbol{N}(\boldsymbol{q},\hat{\boldsymbol{w}}) \tag{23}$$

Note that only actuated joints can be compensated for lower-mobility robots. By extracting $\delta\boldsymbol{\rho}_a$ from $\delta\boldsymbol{\rho}$, the nominal joint variables can be modified as

$$\boldsymbol{q}_m=\boldsymbol{q}-\delta\boldsymbol{\rho}_a \tag{24}$$

## V. Simulation Study

Simulation is carried out on a typical 6-DOF serial robot FANUC Robot M-1000*i*A to verify the effectiveness of the proposed calibration method. As shown in Fig.3, it is a large and heavy payload robot with a wide motion range and 1000kg payload capacity. Due to a high rigidity design, it is available for machining such as drilling and milling.

Figure 3. The CAD model of FANUC Robot M-1000*i*A.

The error model of the actual robot expressed by (13) can be used to generate the pose error. A group of error parameters $\delta\boldsymbol{\zeta}$, $\delta\boldsymbol{\xi}$, $\delta\boldsymbol{q}$ is randomly set in which all the elements conform to normal distributions with zero mean and standard deviation of 0.5mm. By locating three target points on the end effector, position errors of them can be calculated by (18). One hundred configurations are randomly selected in the task space for calibration, in which half of them are used for training and another half of them are used for validation. Since the real measuring process will be inevitably influenced by various factors such as uncertainty of measuring equipment repeatability of the robot and measuring noise, it is necessary to add Gaussian noise to the position errors of target points. The standard deviation of the noise is chosen as 0.05mm.

Fig. 4 shows the training process of the neural network. In order to avoid over-fitting, the convergence of the training process is usually judged on the validation subset. As we can see, the mean squared error on the validation subset will not decrease after 11 iterations.

With the trained neural network, error compensation is conducted. As comparisons, the simulation based on local POE formula [8] is also carried out on the studied robot.

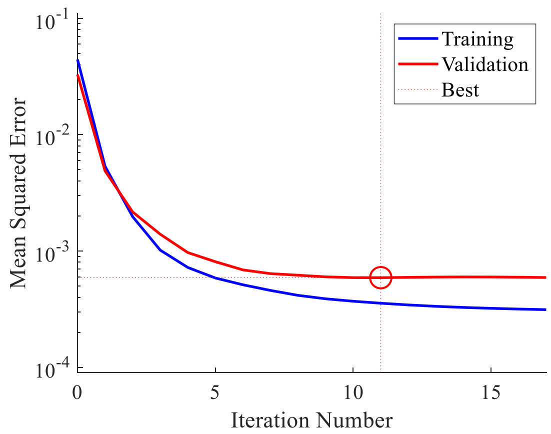


Figure 4. The training process of the neural network.

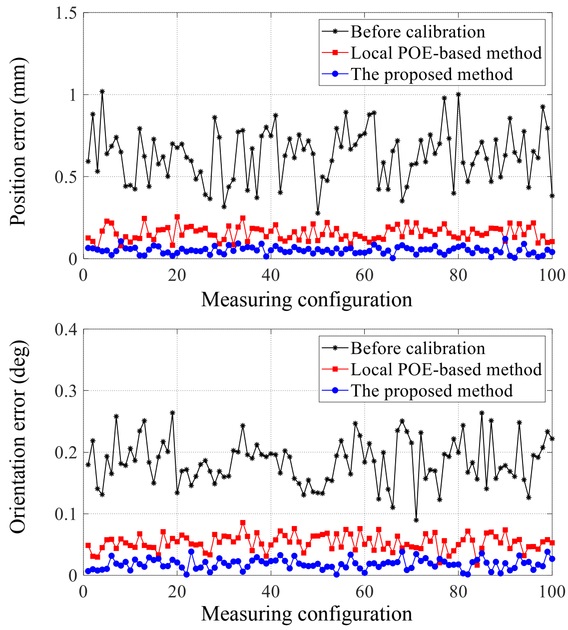


Figure 5. Pose errors before and after calibration.

The pose (position and orientation) errors before and after calibration in one hundred configurations, including both training and validation subsets, are calculated and plotted in Fig. 5. Compared with those before calibration, the maximum values of the position and orientation errors are reduced by 91.90% and 90.21% when using the proposed method, whereas the use of local POE-based method delivers 77.12% and 76.63%. The proposed method also reduces the standard deviations of pose errors more than local POE-based method does. Therefore, we can conclude that the proposed method behaves better than local-POE based method in improving pose accuracy.

## VI. Conclusion and Future Work

A novel calibration method for serial robots is proposed in this paper to improve pose accuracy. The novelty is to treat the actual robot as an equivalent one only with joint motion errors. Based on the connection between the actual and equivalent robots, joint motion errors are revealed to be configuration-dependent variables. Through integration of modified local POE formula and artificial neural networks, the combined error model of the equivalent robot is established, which incorporates the virtues of model-based methods and data-driven methods. Simulation results also prove the superiority of the proposed method.

Although it is general and systematic enough to calibrate any serial robot, there are two problems remaining to be tackled. The first issue is how to apply it to parallel robots, especially for those with lower mobility. Instantaneous screw theory and dual space theory may be powerful mathematical tools for modeling parallel robots. Another issue is how to improve the calibration efficiency since the random selection strategy requires a huge number of measuring configurations.

## Acknowledgment

Y. Song would like to thank China Scholarship Council for funding support (No. 201906250074).

## References

[1] A. Verl, A. Valente, S. Melkote, C. Brecher, E. Ozturk, and L. T. Tunc, “Robots in machining,” *CIRP Annals*, vol. 68, no. 2, pp. 799–822, 2019.

[2] X. Zhao, B. Tao, S. Han, and H. Ding, “Accuracy analysis in mobile robot machining of large-scale workpiece,” *Robotics and Computer-Integrated Manufacturing*, vol. 71, no. 3, p. 102153, 2021.

[3] Z. Li, S. Li, and X. Luo, “An overview of calibration technology of industrial robots,” *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 23–36, 2021.

[4] J. Denavit and R. Hartenberg, “A kinematic notation for lower-pair mechanisms based on matrices,” *J. Appl. Mech.*, vol. 22, no. 2, pp. 215–221, 1955.

[5] S. Hayati, “Robot arm geometric link parameter estimation,” in *Proc. 22th IEEE Conf. Decision Control*, 1983, pp. 1477–1483.

[6] H. Stone and A. Sanderson, “A prototype arm signature identification system,” in *Proc. IEEE Conf. Robot. Autom.*, Raleigh, NC, USA, 1987, pp. 175–182.

[7] K. Okamura and F. C. Park, “Kinematic calibration using the product of exponentials formula,” *Robotica*, vol. 14, no. 4, pp. 415–421, 1996.

[8] I. M. Chen, G. Yang, C. T. Tan, and S. H. Yeo, “Local POE model for robot kinematic calibration,” *Mechanism and Machine Theory*, vol. 36, no. 11–12, pp. 1215–1239, 2001.

[9] G. Chen, H. Wang, and Z. Lin, “Determination of the identifiable parameters in robot calibration based on the POE formula,” *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1066–1077, 2014.

[10] T. Huang, D. Zhao, F. Yin, W. Tian, and D. G. Chetwynd, “Kinematic calibration of a 6-DOF hybrid robot by considering multicollinearity in the identification Jacobian,” *Mechanism and Machine Theory*, vol. 131, pp. 371–384, 2019.

[11] Z. Jiang, M. Huang, X. Tang, B. Song, and Y. Guo, “Elasto-geometrical calibration of six-DOF serial robots using multiple identification models,” *Mechanism and Machine Theory*, vol. 157, p. 104211, 2021.

[12] S. Cao, Q. Cheng, Y. Guo, W. Zhu, H. Wang, and Y. Ke, “Pose error compensation based on joint space division for 6-DOF robot manipulators,” *Precision Engineering*, vol. 74, no. 11, pp. 195–204, 2022.

[13] D. Chen, P. Yuan, T. Wang, Y. Cai, and L. Xue, “A compensation method for enhancing aviation drilling robot accuracy based on co-kriging,” *International Journal of Precision Engineering and Manufacturing*, vol. 19, no. 8, pp. 1133–1142, 2018.

[14] Y. Bai and D. Wang, “Calibrate parallel machine tools by using interval type-2 fuzzy interpolation method,” *International Journal of Advanced Manufacturing Technology*, vol. 93, no. 9–12, pp. 3777–3787, 2017.

[15] G. Alici and B. Shirinzadeh, “A systematic technique to estimate positioning errors for robot accuracy improvement using laser interferometry based sensing,” *Mechanism and Machine Theory*, vol. 40, no. 8, pp. 879–906, 2005.

[16] X. Zhong, J. M. Lewis, and F. L. N-Nagy, “Inverse robot calibration using artificial neural networks,” *Engineering Applications of Artificial Intelligence*, vol. 9, no. 1, pp. 83–93, 1996.

[17] H. N. Nguyen, J. Zhou, and H. J. Kang, “A calibration method for enhancing robot accuracy through integration of an extended Kalman filter algorithm and an artificial neural network,” *Neurocomputing*, vol. 151, no. 3, pp. 996–1005, 2015.

[18] H. Liu, T. Huang, and D. G. Chetwynd, “A general approach for geometric error modeling of lower mobility parallel manipulators,” *Journal of Mechanisms and Robotics*, vol. 3, no. 2, p. 021013, 2011.

[19] I.A. Basheer and M. Hajmeer, “Artificial neural networks: fundamentals, computing, design, and application,” *Journal of Microbiological Methods*, vol. 43, pp. 3–31, 2013.

[20] J. Schmidhuber, “Deep Learning in neural networks: An overview,” Neural Networks, vol. 61, pp. 85–117, 2015.