Calibration of a Stewart platform by designing a robust joint compensator with artificial neural networks

Yongbin Song [a], Wenjie Tian [b,*], Yanling Tian [a], Xianping Liu [a]

[a] *School of Engineering, The University of Warwick, Coventry CV4 7AL, UK*

[b] *School of Marine Science and Technology, Tianjin University, Tianjin 300072, China*

Abstract: By taking a Stewart platform as an example, this paper presents a novel calibration method by designing a robust joint compensator based on artificial neural networks. In this method, the pose error arising from various time-independent error sources is treated as that produced only by configuration-dependent joint motion errors equivalently, thus allowing the static pose error to be eliminated by directly correcting the nominal joint variables. Hence, the calibration procedure can be implemented in three successive steps: (1) acquisition of necessary joint corrections with point measurement at finite configurations considering near singularity problems, (2) approximation of the function between joint corrections and nominal joint variables using feedforward neural networks with coupled/decoupled architectures, and (3) design of a joint compensator embedded in the numerical control system to realize online real-time error compensation. Experimental results show that the proposed robust compensator based on coupled or decoupled networks can significantly improve the static pose accuracy in comparison with previous methods.

Keywords: Calibration; Joint compensator; Artificial neural network; Stewart platform

1. Introduction

Stewart platform is a typical parallel manipulator which can perform a 6-DOF (degree of freedom) movement [1]. In recent years, it has played an essential role in many applications, such as flight simulators, robotic machining, and underwater research [2]. Static pose accuracy is one of the most

* Corresponding author

Tel: +86 15822450489

E-mail address: wenjietian@tju.edu.cn

ORCID: 0000-0002-7853-8463

significant performance specifications for this manipulator. Since the actual robot is inevitably contaminated by manufacturing tolerances, link offsets, and other error sources, the resulting poses driven by nominal joint variables are different from the desired ones. Calibration is a highly cost-effective method to enhance the pose accuracy through software without increasing the manufacturing cost [3-5].

1.1. literature review

Over the past few decades, there has been a large volume of studies focusing on manipulator calibration. The available methods can be generally classified into two categories: model-based and data-driven methods.

The model-based methods aim to find the actual forward kinematics by identifying exact kinematic parameters. Specifically, the linearized error model between the pose error and kinematic errors is derived by differentiating the forward kinematics, and then it can be used to estimate actual kinematic parameters iteratively using nominal kinematic parameters as initial values in the least square sense. To meet the requirements of completeness, continuity, and minimality, the modified D-H model [6,7] and the product of exponentials (POE) formula [8,9] are the most popular model-based methods. The former is based on homogeneous transformation matrices, whereas the latter is based on the classical screw theory. In addition, the unit dual quaternion is an alternative method based on the transformation and representation of joint axes, which has a more compact and consistent form [10]. The above methods can be applied to construct forward kinematics for serial robots conveniently. However, parallel manipulators pose more challenges in modeling due to a large number of kinematic parameters in all limbs and the complicated closed-loop structure. Zhuang and Roth [11] proposed a simplified calibration method for the Stewart platform by differentiating the closed-loop vector equations. The limitation is that some error sources are neglected due to the assumption that the universal and spherical joints are perfectly machined. To solve this problem, Chen et al. [12] proposed a complete, minimal, and continuous error modeling method for parallel robots based on the POE formula. Liu et al. [13] also reported a general approach using instantaneous screw theory. Although model-based methods have definite physical meanings, the error modeling process and redundant parameter elimination are complicated. Moreover, it is difficult to construct and calculate forward/inverse kinematics considering all geometric error sources. Several unmodeled nongeometric error sources, such as pitch errors and elastic deformation, also restrict the calibration accuracy [14].

On the other hand, the data-driven methods do not need to model error sources but treat the relationship between the pose error and the configuration as a "black box". They retain the nominal kinematic model and require the "black box" to be approximated using spatial data interpolation (fitting) techniques. The interpolation methods divide the task space into several cubic cells, measure position errors on the grid points in each cell, and then store them in memory for interpolation and compensation of position errors at any possible configurations. These interpolation methods include the inverse distance weighted (IDW) method [15], the Kriging method [16,17], and the fuzzy method [18,19]. Although they are feasible for different kinds of parallel robots with complex topology, they would require a huge size of memory space if high calibration accuracy is needed. Besides, they ignore the influence of the orientation variation and only improve the position accuracy due to the dimension limitation of spatial interpolation. To address this problem, some fitting approaches such as Fourier polynomials [20], support vector machine (SVM) [21], and artificial neural networks (ANNs) [22,23] have become available. In these approaches, regression models are trained using nominal joint variables as inputs and measured pose errors as outputs. They can solve the high-dimensional curve-fitting problem well and only need to store coefficients/weights instead of all measured data. The data-driven approaches are not limited by imprecise modeling of all error sources but only depend on the fitting capability of the approximation functions. However, since no prior knowledge of the robot is put to use in the data-driven methods, the required measuring configurations are usually more than model-based methods to achieve the same level of calibration accuracy.

Even though high prediction accuracy of pose errors can be guaranteed based on the above methods, some problems still need to be tackled when it comes to error compensation. For instance, traditional iterative algorithms based on Jacobian might be unsuitable for online real-time compensation due to high computational expense, and might even break down near singularity zones [24]. To avoid these issues, Whitney and Shamma [25] reported an alternative data-driven calibration method by approximating the joint corrections obtained offline directly rather than pose errors. Nevertheless, the measurement process is quite time-consuming because the joint corrections need to be acquired by compensating for pose errors manually.

1.2. The main contribution

Motivated by the practical needs for high calibration accuracy and robust online error compensation,

this paper presents a novel calibration method for the Stewart platform. Treating the pose error arising from various error sources as that produced only by configuration-dependent joint motion errors equivalently, this method can combine the virtues of model-based and data-driven methods.

In this method, the simplified linearized error model, which only considers equivalent joint motion errors, is deduced based on the screw theory. The model is complete and nonredundant since six and only six parameters need to be determined for a given configuration. The prior knowledge of the robot can be utilized effectively to guarantee the stability of joint motion errors. However, unlike the traditional model-based methods, the equivalent joint motion errors rely on the current configuration, and thus they cannot be assumed to be constant. Instead, they should be seen as complicated nonlinear functions of configuration expressed by nominal joint variables. Thus, data-driven methods such as ANNs can be applied to describe the relationship between them due to the high approximation capability. Two architectures of ANNs are designed, and their performances are compared in the experimental study. Although configuration-dependent joint motion errors do not have any physical meanings, we can achieve the online error compensation simply by correcting the nominal joint variables with them instead of altering the kinematic parameters.

The rest of this paper is organized as follows. After a brief description of the Stewart platform in Section 2, Section 3 proposes the methodology that can be implemented in three steps: (1) acquisition of joint corrections with point measurement taking into account near singular configurations, (2) approximation of the function between joint corrections and nominal joint variables by training coupled or decoupled ANNs, and (3) design of the embedded joint compensator based on trained ANNs to achieve online real-time compensation. Section 4 carries on calibration experiments on a prototype to demonstrate the effectiveness of the proposed method, before conclusions are drawn in Section 5.

2. System description

Fig.1 shows the schematic diagram of the Stewart platform. It is composed of a fixed base, a moving platform, and six spatial UPS limbs. In order to establish the standardized mathematical model, we number each UPS limb as limb i ($i = 1 \sim 6$). Here, U and S represent universal and spherical joints respectively, and P denotes actuated prismatic joint. For convenience, universal/spherical joints can be treated as two/three revolute joints with the joint axes intersecting at a common point.

Let $\boldsymbol{s}_{j,i}$ denote the unit vector of the jth joint axis in limb i, which satisfies the following relationship.

$$\boldsymbol{s}_{1,i} \perp \boldsymbol{s}_{2,i}, \quad \boldsymbol{s}_{2,i} \perp \boldsymbol{s}_{3,i}, \quad \boldsymbol{s}_{3,i} = \boldsymbol{s}_{4,i}, \quad \boldsymbol{s}_{4,i} \perp \boldsymbol{s}_{5,i}, \quad \boldsymbol{s}_{5,i} \perp \boldsymbol{s}_{6,i}, \quad i = 1 \sim 6 \tag{1}$$

Several reference points are defined as follows without regard to error sources. B_i and A_i denote the center of U and S joint in limb i, respectively. B and A represent the center of the base and the platform, respectively. A base frame $\mathcal{K}_B$ is established at point B with its z-axis normal to the base plane, and a body-fixed frame $\mathcal{K}_A$ is located at point A with its z-axis perpendicular to the platform plane. To evaluate the error twist about point A conveniently, an instantaneous frame $\mathcal{K}_A'$ is placed at point A with its coordinate axes parallel with those of $\mathcal{K}_B$. It should be noted that $\mathcal{K}_A'$ and $\mathcal{K}_A$ are coincident at the reference configuration.

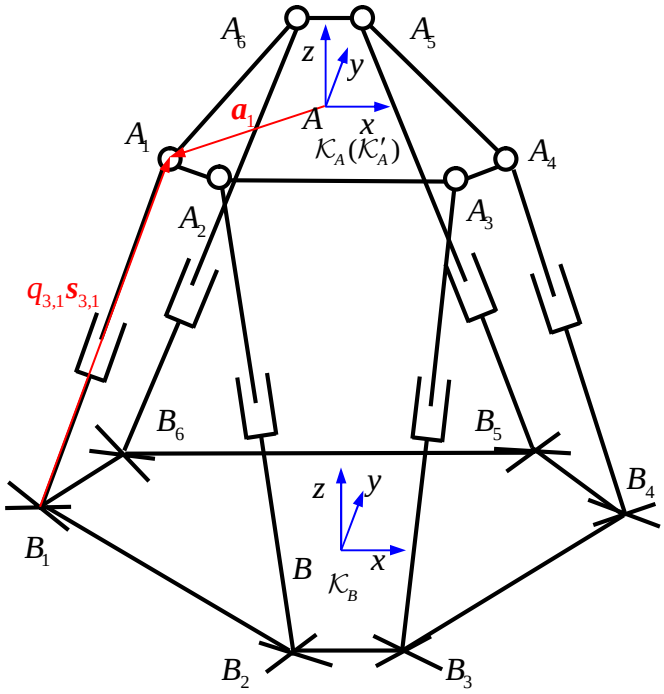

Fig. 1. The schematic diagram of the Stewart platform.

3. Methodology

In this section, the proposed calibration methodology is described by taking the Stewart platform as an example. It generally includes three steps: acquisition of joint corrections, approximation of the joint correction function, and design of the joint compensator.

3.1. Acquisition of the joint corrections with point measurement

This subsection describes how to acquire the necessary joint corrections robustly through point measurement. Firstly, the linear relationship between the position errors of target points and the joint motion errors is established based on the screw theory. Then, the latter are solved iteratively using the L-M algorithm with the measured information of the former. Lastly, the joint corrections are determined by the opposite of the cumulative joint motion errors.

Drawing upon the screw theory-based error modeling method [13], the pose error twist of the end effector can be described as a linear combination of six actuated joint basis vectors at a given configuration. The corresponding configuration-dependent coordinates are termed as joint motion errors herein. At a deeper level, the pose error arising from various error sources is treated as that produced only by configuration-dependent joint motion errors equivalently. Therefore, the relationship between the pose error twist and joint motion errors for the Stewart platform is derived as follows.

Because all limbs share the same moving platform, the pose error twist $\boldsymbol{\xi}_t$ about point A can be expressed as

$$\boldsymbol{\xi}_t = \sum_{j=1}^{6} \rho_{j,i} \hat{\boldsymbol{\xi}}_{ta,j,i}, \quad i = 1 \sim 6 \tag{2}$$

with

$$\boldsymbol{\xi}_t = \begin{pmatrix} \boldsymbol{\delta}_A \\ \boldsymbol{\varepsilon} \end{pmatrix}, \quad \hat{\boldsymbol{\xi}}_{ta,1,i} = \begin{pmatrix} \left(\boldsymbol{a}_i - q_{3,i}\boldsymbol{s}_{3,i}\right) \times \boldsymbol{s}_{1,i} \\ \boldsymbol{s}_{1,i} \end{pmatrix}, \quad \hat{\boldsymbol{\xi}}_{ta,2,i} = \begin{pmatrix} \left(\boldsymbol{a}_i - q_{3,i}\boldsymbol{s}_{3,i}\right) \times \boldsymbol{s}_{2,i} \\ \boldsymbol{s}_{2,i} \end{pmatrix}, \quad \hat{\boldsymbol{\xi}}_{ta,3,i} = \begin{pmatrix} \boldsymbol{s}_{3,i} \\ \boldsymbol{0} \end{pmatrix}$$

$$\hat{\boldsymbol{\xi}}_{ta,4,i} = \begin{pmatrix} \boldsymbol{a}_i \times \boldsymbol{s}_{4,i} \\ \boldsymbol{s}_{4,i} \end{pmatrix}, \quad \hat{\boldsymbol{\xi}}_{ta,5,i} = \begin{pmatrix} \boldsymbol{a}_i \times \boldsymbol{s}_{5,i} \\ \boldsymbol{s}_{5,i} \end{pmatrix}, \quad \hat{\boldsymbol{\xi}}_{ta,6,i} = \begin{pmatrix} \boldsymbol{a}_i \times \boldsymbol{s}_{6,i} \\ \boldsymbol{s}_{6,i} \end{pmatrix}, \quad \boldsymbol{a}_i = \overrightarrow{AA_i}, \quad q_{3,i} = \left\| \overrightarrow{B_i A_i} \right\|$$

where $\boldsymbol{\delta}_A \in \mathbb{R}^3$ and $\boldsymbol{\varepsilon} \in \mathbb{R}^3$ represent the position error of point A and the orientation error of the platform, respectively; $\hat{\boldsymbol{\xi}}_{ta,j,i}$ and $\rho_{j,i}$ represent the unit twist of jth ($j = 1 \sim 6$) joint in limb i ($i = 1 \sim 6$) and its corresponding joint motion error. Because only actuated joints can be corrected via compensation, only actuated joint motion errors $\rho_{3,i}$ need to be retained by utilizing dual properties of the twist space and the wrench space. Considering that a wrench of actuation only does virtual work on the permitted twist caused by itself, we take inner products on both sides of Eq. (2) with $\hat{\boldsymbol{\xi}}_{wa,3,i}$ and

rewrite them into a matrix form.

$$\boldsymbol{W}^{\mathrm{T}}\boldsymbol{\xi}_t = \boldsymbol{\rho}\,,\quad \boldsymbol{\xi}_t = \boldsymbol{T}\boldsymbol{\rho} \tag{3}$$

with

$$\boldsymbol{W} = \begin{bmatrix} \hat{\boldsymbol{\xi}}_{wa,3,1} & \hat{\boldsymbol{\xi}}_{wa,3,2} & \cdots & \hat{\boldsymbol{\xi}}_{wa,3,6} \end{bmatrix},\quad \boldsymbol{\rho} = \begin{pmatrix} \rho_{3,1} & \rho_{3,2} & \cdots & \rho_{3,6} \end{pmatrix}^{\mathrm{T}},\quad \boldsymbol{T} = \boldsymbol{W}^{-\mathrm{T}}$$

$$\hat{\boldsymbol{\xi}}_{wa,3,i} = \begin{pmatrix} \boldsymbol{s}_{3,i} \\ \boldsymbol{a}_i \times \boldsymbol{s}_{3,i} \end{pmatrix},\quad i = 1 \sim 6$$

where $\boldsymbol{T} \in \mathbb{R}^{6\times 6}$ and $\boldsymbol{W} \in \mathbb{R}^{6\times 6}$ denote the motion Jacobian and the force Jacobian, respectively; $\hat{\boldsymbol{\xi}}_{wa,3,i}$ denotes the unit wrench of actuation provided by limb i, and $\boldsymbol{\rho} \in \mathbb{R}^6$ represents the vector of actuated joint motion errors. It is obvious that $\boldsymbol{\xi}_t$ should be measured in order to acquire $\boldsymbol{\rho}$ at a given configuration, but there are few measuring devices that can obtain the orientation error $\boldsymbol{\varepsilon}$ directly. Therefore, position errors of at least three noncolinear points are measured to replace $\boldsymbol{\xi}_t$. The relationship between them can be expressed as

$$\boldsymbol{\zeta} = \boldsymbol{P}\boldsymbol{\xi}_t \tag{4}$$

with

$$\boldsymbol{\zeta} = \boldsymbol{r}_{Pm} - \boldsymbol{r}_{Pc} = \begin{pmatrix} \boldsymbol{\delta}_{P,1} \\ \boldsymbol{\delta}_{P,2} \\ \boldsymbol{\delta}_{P,3} \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_{Pm,1} - \boldsymbol{r}_{Pc,1} \\ \boldsymbol{r}_{Pm,2} - \boldsymbol{r}_{Pc,2} \\ \boldsymbol{r}_{Pm,3} - \boldsymbol{r}_{Pc,3} \end{pmatrix},\quad \boldsymbol{P} = \begin{pmatrix} \boldsymbol{P}_1 \\ \boldsymbol{P}_2 \\ \boldsymbol{P}_3 \end{pmatrix},\quad \boldsymbol{P}_j = \begin{bmatrix} \boldsymbol{I}_3 & \left[\boldsymbol{p}_j \times\right] \end{bmatrix},\quad \boldsymbol{p}_j = \overrightarrow{P_j A},\quad j = 1 \sim 3$$

where $\boldsymbol{\delta}_{P,j} \in \mathbb{R}^3$ represents the position error of target point P_j; $\boldsymbol{r}_{Pm,j} \in \mathbb{R}^3$ and $\boldsymbol{r}_{Pc,j} \in \mathbb{R}^3$ represent the measured and computed position vectors of $\overrightarrow{BP_j}$, respectively; $\boldsymbol{I}_3 \in \mathbb{R}^{3\times 3}$ denotes a third order identity matrix; $\left[\boldsymbol{p}_j \times\right] \in \mathbb{R}^{3\times 3}$ denotes a skew matrix of $\boldsymbol{p}_j$. Substituting Eq. (3) into Eq. (4) leads to

$$\boldsymbol{\zeta} = \boldsymbol{P}\boldsymbol{T}\boldsymbol{\rho} \tag{5}$$

Suppose $\boldsymbol{r}_{Pm}$ has been measured by the external point measuring equipment and $\boldsymbol{r}_{Pc}$ has been computed through the nominal forward kinematics, $\boldsymbol{\rho}$ can be obtained directly by solving Eq. (5), which is an overdetermined equation due to inevitable measurement noise. Considering that target points P_j ($j = 1 \sim 3$) are not collinear, $\boldsymbol{P} \in \mathbb{R}^{9\times 6}$ is certain to be column full rank. However, the motion/force

Jacobian $\boldsymbol{T}/\boldsymbol{W}$ may be ill-conditioned or rank deficient in the vicinity of singular configurations. Therefore, solving Eq. (5) through the ordinary least square method could lead to extremely sensitive computed joint motion errors $\boldsymbol{\rho}$. In order to guarantee $\boldsymbol{\rho}$ as small as possible, the problem is then to determine $\boldsymbol{\rho}$ such that the following loss function is minimized.

$$\Phi(\boldsymbol{\rho}) = \left\| \boldsymbol{\zeta} - \boldsymbol{P}\boldsymbol{T}\boldsymbol{\rho} \right\|^2 + \lambda \left\| \boldsymbol{\rho} \right\|^2 \tag{6}$$

where the first term is the sum of squared residuals of position errors, and the second term is the regularization term to avoid large changes of $\boldsymbol{\rho}$. The regularization parameter λ can be adjusted to strike a balance between these two terms. Since both terms in the loss function have quite the same units of length, it is unnecessary to use the weighted least squares technique to modify the loss function. $\boldsymbol{\rho}$ can then be solved by

$$\boldsymbol{\rho} = \left((\boldsymbol{P}\boldsymbol{T})^{\mathrm{T}} (\boldsymbol{P}\boldsymbol{T}) + \lambda \boldsymbol{I}_6 \right)^{-1} (\boldsymbol{P}\boldsymbol{T})^{\mathrm{T}} \boldsymbol{\zeta} \tag{7}$$

where $\boldsymbol{I}_6 \in \mathbb{R}^{6\times 6}$ denotes a sixth order identity matrix. The choice of λ depends on how well-conditioned the matrix $\boldsymbol{P}\boldsymbol{T} \in \mathbb{R}^{9\times 6}$ is. The condition number of the rectangular matrix is defined as the ratio between the maximum and minimum singular values. Specifically, if $\mathrm{cond}(\boldsymbol{P}\boldsymbol{T})$ is larger than c, λ is set to λ_0, otherwise λ is set to zero. Here, c and λ_0 are the specified condition number and regularization parameter, respectively. It is worth pointing out that $\boldsymbol{P}\boldsymbol{T}$ with large condition number is only attributed to configuration singularity since it is a dimensionless matrix that avoids the nonessential ill-conditioned problem caused by dimensional inconsistency.

Note that the linearized error model expressed by Eq. (5) is valid only when deviations between the actual pose and the nominal pose are small enough. While in most cases, this assumption is slightly unreasonable. Therefore, the regularization solution expressed by Eq. (7) can be further improved in an iterative manner (see Fig. 2), where $\boldsymbol{f}(\bullet)$ represents the nominal forward kinematics. The iterative version is also known as Levenberg-Marquardt (L-M) algorithm [26]. Let $\boldsymbol{q}^{(l)}$ denote the joint variables containing joint motion errors in the l th iteration, and initialize $\boldsymbol{q}^{(0)}$ using the nominal value $\boldsymbol{q}$ calculated by nominal inverse kinematics. After $\boldsymbol{\rho}^{(l)}$ is determined by Eq. (7), the joint variables can

then be updated as

$$\boldsymbol{q}^{(l+1)} = \boldsymbol{q}^{(l)} + \boldsymbol{\rho}^{(l)}, \quad l = 0,1,2,\cdots \tag{8}$$

In each iteration, $\boldsymbol{P}$, $\boldsymbol{T}$ and $\boldsymbol{\zeta}$ in Eq. (7) are updated by substituting $\boldsymbol{q}^{(l)}$ with $\boldsymbol{q}^{(l+1)}$. The iteration process will not terminate until the norm of $\boldsymbol{\rho}^{(l)}$ is below the specified threshold κ. If L is the last iteration, the necessary joint corrections, which are also the opposite of the cumulative joint motion errors, can be calculated as

$$\Delta\boldsymbol{q} = -\left(\boldsymbol{q}^{(L)} - \boldsymbol{q}^{(0)}\right) = \boldsymbol{q}^{(0)} - \boldsymbol{q}^{(L)} \tag{9}$$

Although the algorithm may need complicated numerical calculations due to the forward kinematics, it can be implemented offline, fortunately. The robustness lies in that it guarantees the stability and smoothness of the joint corrections at all configurations, especially at near singular configurations.

Input: point measurement information $\boldsymbol{r}_{Pm}$, nominal joint variables $\boldsymbol{q}$
Output: joint corrections $\Delta\boldsymbol{q}$
Initialization: $l := 0$, $\boldsymbol{q}^{(l)} := \boldsymbol{q}$
Do
$\boldsymbol{r}_{Pc} = \boldsymbol{f}\left(\boldsymbol{q}^{(l)}\right)$, $\boldsymbol{\zeta} = \boldsymbol{r}_{Pm} - \boldsymbol{r}_{Pc}$, $\boldsymbol{P}\left(\boldsymbol{q}^{(l)}\right)$, $\boldsymbol{T}\left(\boldsymbol{q}^{(l)}\right)$
If ($\mathrm{cond}\left(\boldsymbol{PT}\right) > c$)
$\lambda := \lambda_0$
Else
$\lambda := 0$
End if
$\boldsymbol{\rho}^{(l)} = \left(\left(\boldsymbol{PT}\right)^{\mathrm{T}}\left(\boldsymbol{PT}\right) + \lambda\boldsymbol{I}_6\right)^{-1}\left(\boldsymbol{PT}\right)^{\mathrm{T}}\boldsymbol{\zeta}$, $\boldsymbol{q}^{(l+1)} := \boldsymbol{q}^{(l)} + \boldsymbol{\rho}^{(l)}$, $l := l+1$
Loop until ($\left\|\boldsymbol{\rho}^{(l)}\right\| < \kappa$)
$L := l$, $\Delta\boldsymbol{q} = \boldsymbol{q}^{(0)} - \boldsymbol{q}^{(L)}$

Fig. 2. Acquisition of joint corrections based on the L-M algorithm.

3.2. Approximation of the joint correction function based on ANNs

In this subsection, we investigate how to describe the complicated nonlinear function between the nominal joint variables (inputs) and the joint corrections (outputs) with ANNs, for the purpose of

predicting joint corrections in any possible configurations.

It is widely known that multilayered feedforward ANNs 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 [27]. Moreover, they are easy to be implemented and programmed in the numerical control (NC) system [28], which benefits the design of the embedded joint compensator. It is reasonable to adopt simple three-layered feedforward ANNs herein since the joint corrections do not have sharp fluctuations.

The principle of the feedforward ANN is briefly introduced here. It is comprised of densely interconnected nodes that are able to perform massive nonlinear computations. Generally, a three-layered network involves an input layer, a hidden layer, and an output layer. The nodes in the hidden layer are connected with those in the input/output layer through adjustable connection weights. For a given set of inputs, the network can compute the corresponding outputs via a feedforward solution with initial weights. Then, these weights can be modified through the backward propagation of the errors to make the computed outputs as close as possible to the observed outputs. In other words, network training aims to minimize the error surface, which is a nonlinear function of connection weights. Interestingly, the training process shows a similar pattern to the solution of nonlinear least squares, and thus many practical nonlinear regression methods could also be used for training.

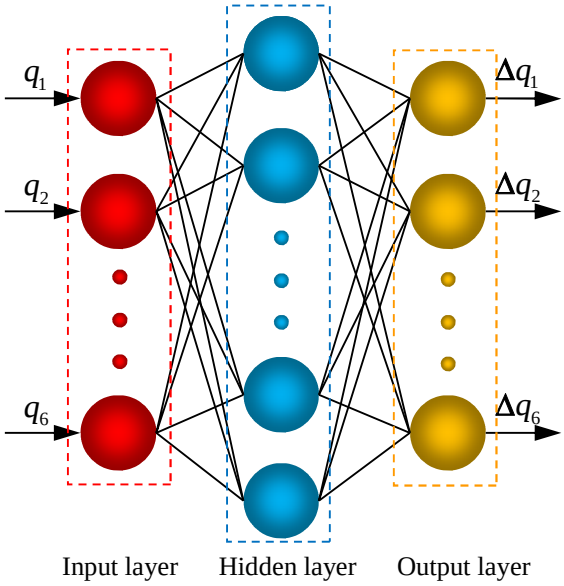

Fig. 3. The architecture of the coupled network.

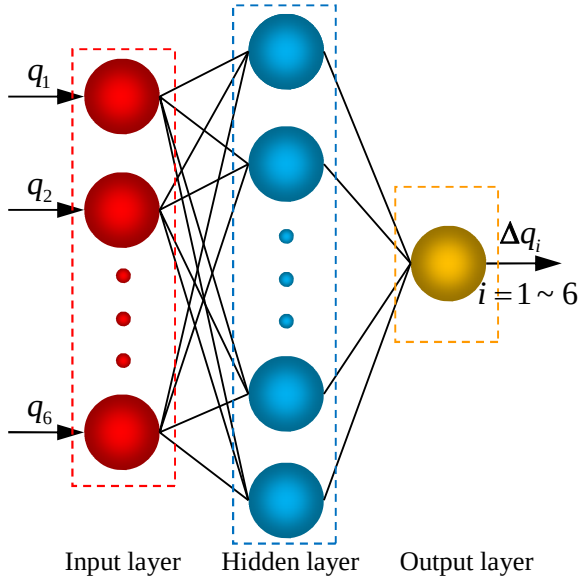

Fig. 4. The architecture of the decoupled network.

As shown in Fig.3 and Fig.4, we propose two architectures of coupled and decoupled networks. In the coupled network, all six joint corrections are assigned to be outputs, and thus only one network needs to be trained. As to the decoupled network, each joint correction is selected to be the output individually, and thus six decoupled networks need to be constructed for training independently. For convenience, the training procedures will be described by taking the coupled network as an example in the following. The training efficiency, optimal architecture, and fitting capability of the two networks will be compared in Section 4.

For a general node in the network, the output of it can be calculated as

$$y_i^t = \sigma\left(z_i^t\right) = \sigma\left(\sum_{j=1}^{n} w_{ij}^t x_j^{t-1} + b_i^t\right), \quad t = 2,3 \tag{10}$$

where x_j^{t-1} denotes the output of j th node in the $t-1$ layer, and also one of the inputs of i th node in the t layer; y_i^t denotes the output of i th node in the t layer; w_{ij}^t denotes the connection weight between the j th node in the $t-1$ layer and i th node in the t layer; b_i^t denotes the bias of the i th node in the t layer; $\sigma(\bullet)$ denotes the activation function which transforms the weighted sum of all inputs to the output. As a rule of thumb, tan-sigmoid and linear function can be selected as the activation function of the hidden layer and output layer respectively, that is

$$\sigma_H(z)=\frac{1-e^{-2z}}{1+e^{-2z}} \tag{11}$$

$$\sigma_O(z)=kz \tag{12}$$

To prevent the training process impeded due to extremely large input data, it is essential to normalize the nominal actuated joint variables as

$$q_i := \frac{q_i - q_{i,\text{mean}}}{q_{i,\max} - q_{i,\min}}, \quad i = 1 \sim 6 \tag{13}$$

where $q_{i,\text{mean}}$, $q_{i,\max}$ and $q_{i,\min}$ are the mean, maximum and minimum values of the i th actuated joint motion range, respectively. For the sake of brevity, the function of the network can be expressed in an implicit form as

$$\Delta \boldsymbol{q} = \boldsymbol{N}(\boldsymbol{q}, \boldsymbol{w}) \tag{14}$$

where $\boldsymbol{q} \in \mathbb{R}^6$ and $\Delta \boldsymbol{q} \in \mathbb{R}^6$ denote the input and the output of the network respectively; $\boldsymbol{w} \in \mathbb{R}^N$ denotes the combined weight vector containing all weights and biases. Suppose that we have taken point measurements and calculated corresponding joint corrections $\Delta \boldsymbol{q}$ in K configurations, we can acquire K training samples of input/output pairs. Then, the loss function can be defined as the mean square error between observed joint corrections $\Delta \boldsymbol{q}_m$ from point measurement and the computed joint corrections $\Delta \boldsymbol{q}_c$ from the network for all training samples.

$$E(\boldsymbol{w}) = \frac{1}{K}\sum_{k=1}^{K}\left\|\Delta \boldsymbol{q}_{m,k} - \Delta \boldsymbol{q}_{c,k}\right\|^2 = \frac{1}{K}\sum_{k=1}^{K}\left\|\Delta \boldsymbol{q}_{m,k} - \boldsymbol{N}(\boldsymbol{q}_k, \boldsymbol{w})\right\|^2 \tag{15}$$

In order to minimize the loss function, the L-M algorithm can be used to search for the optimum weight vector iteratively [26]. The L-M algorithm incorporates the merits of both the fast training speed of the Gauss-Newton algorithm and the guaranteed convergence of the gradient descent method, in which the regularization parameter can be adjusted to make the trade-off [28]. In each iteration step, the weight correction vector $\Delta \boldsymbol{w}$ can be calculated by

$$\Delta \boldsymbol{w} = \left(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J} + \mu \boldsymbol{I}_N\right)^{-1}\boldsymbol{J}^{\mathrm{T}}\boldsymbol{e} \tag{16}$$

with

$$\boldsymbol{J}=\begin{bmatrix}\dfrac{\mathrm{d}\boldsymbol{N}\left(\boldsymbol{q}_1,\boldsymbol{w}\right)}{\mathrm{d}\boldsymbol{w}}\\ \vdots\\ \dfrac{\mathrm{d}\boldsymbol{N}\left(\boldsymbol{q}_K,\boldsymbol{w}\right)}{\mathrm{d}\boldsymbol{w}}\end{bmatrix},\quad \boldsymbol{e}=\begin{bmatrix}\Delta\boldsymbol{q}_{m,1}-\boldsymbol{N}\left(\boldsymbol{q}_1,\boldsymbol{w}\right)\\ \vdots\\ \Delta\boldsymbol{q}_{m,K}-\boldsymbol{N}\left(\boldsymbol{q}_K,\boldsymbol{w}\right)\end{bmatrix}$$

where $\boldsymbol{J}\in\mathbb{R}^{6K\times N}$ is the Jacobian matrix of the network that can be calculated using the chain rule at the current weight vector; μ is the adjustable regularization parameter; $\boldsymbol{I}_N\in\mathbb{R}^{N\times N}$ is a Nth order identity matrix; $\boldsymbol{e}\in\mathbb{R}^{6K}$ is the residual vector. Initializing $\boldsymbol{w}^{(0)}$ using a random vector in which all the components conform to a standard normal distribution, we can update it as

$$\boldsymbol{w}^{(l+1)}=\boldsymbol{w}^{(l)}+\Delta\boldsymbol{w}^{(l)},\quad l=0,1,2,\cdots \tag{17}$$

The training process will proceed until the relative change of the loss functions between two successive iterations reduces to the desired threshold. The weight vector in the last iteration will then be determined as the training result $\hat{\boldsymbol{w}}$. The commonly used evaluation index of the neural network is the root mean square error (RMSE) as

$$\mathrm{RMSE}=\sqrt{\frac{1}{K}\sum_{k=1}^{K}\left\|\Delta\boldsymbol{q}_{m,k}-\boldsymbol{N}\left(\boldsymbol{q}_k,\hat{\boldsymbol{w}}\right)\right\|^2} \tag{18}$$

The above training process is described assuming that the architecture of the network has been given. However, the selection of the optimal number of hidden nodes is an important issue that needs to be considered carefully. Although many empirical formulas have been proposed, the holdout cross-validation method based on the experiment is still one of the most efficient approaches to address this problem. Specifically, the overall K samples are assigned to the training and validation subset. K_1 samples of the training subset are used for training the networks with different numbers of hidden nodes, while K_2 samples of the validation subset are used for evaluating their performances to select the optimal number of hidden nodes.

3.3. Design of the joint compensator for online compensation

On the basis of the trained network, online error compensation can be achieved by designing a compensator embedded in the NC system (see Fig. 5). Firstly, all weights are stored in the memory in advance for calling. For a given continuous path of the moving platform, a sequence of discrete command

pose datasets are generated via rough interpolation. Given the command pose $\boldsymbol{y}$ in the data buffer, the nominal joint variables $\boldsymbol{q}$ are calculated with nominal inverse kinematics. Then, the joint corrections can be predicted as

$$\Delta \boldsymbol{q} = \boldsymbol{N}\left(\boldsymbol{q}, \hat{\boldsymbol{w}}\right) \tag{19}$$

The nominal joint variables can then be corrected as

$$\boldsymbol{q}_m = \boldsymbol{q} + \Delta \boldsymbol{q} \tag{20}$$

The datasets of the corrected joint variables are further produced via fine interpolation and then sent to each actuated prismatic joint to drive the moving platform to complete the desired path.

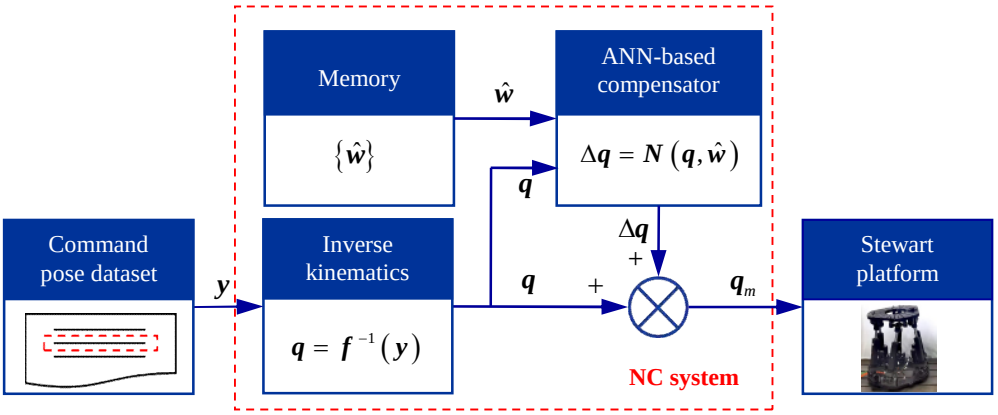

Fig. 5. Work flowchart of the joint compensator embedded in the NC system.

4. Verification

In this section, the experiment is carried out on a Stewart platform prototype to verify the effectiveness of the proposed calibration method. The performances and calibration accuracy of coupled and decoupled networks are evaluated. Comparisons between the proposed method and previous methods are also discussed.

4.1. Experimental setup

As shown in Fig. 6, the radii of the fixed base and the moving platform are 370 mm and 250 mm, respectively. The motion range of each actuated prismatic joint is from 280 mm to 430 mm. The task workspace is a ϕ150mm×100mm cylinder with $\pm 30°$ horizontal tilting angle. According to ISO 9283 [29], the position repeatability and orientation repeatability have been tested as 0.026 mm and 0.008 deg,

respectively.

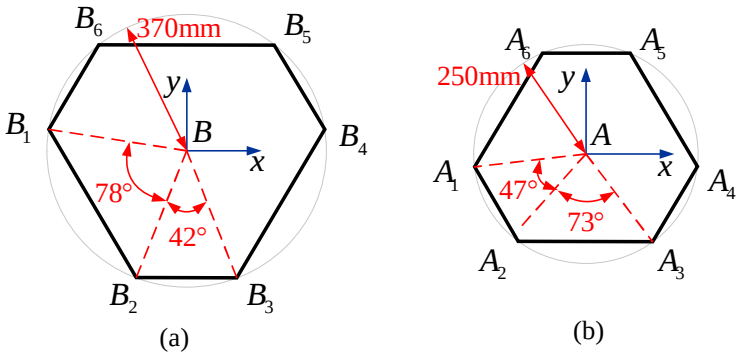

Fig. 6. Dimensions of the Stewart platform. (a) The base. (b) The platform.

As shown in Fig. 7, the position data measuring equipment is a FARO measuring arm with the uncertainty of 0.020 mm. Three measuring cones representing target points P_j ($j = 1 \sim 3$) are located at the upper surface of the moving platform and connected to the moving platform through the thread. Although the nominal locations of measuring cones are known, positional deviations of them should still be determined by the measuring arm aforehand in order to reduce the influences of manufacturing and assembly errors. Besides, three measuring cones are installed on the fixed base to establish the base frame $\mathcal{K}_B$. It should be noted that all subsequent measurement information will be evaluated in $\mathcal{K}_B$.

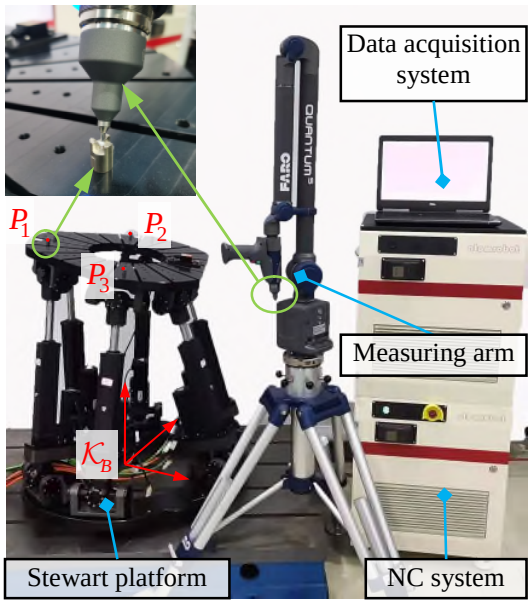

Fig. 7. Experimental setup for calibration.

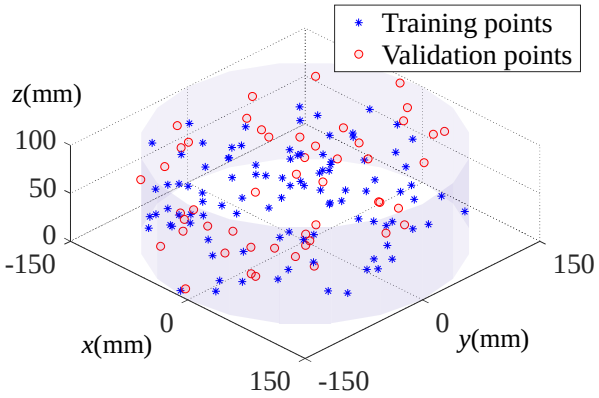

Fig. 8. Distributions of measuring points in the task workspace.

One hundred and fifty configurations are randomly selected on the premise that they are located in the concerned cylinder task workspace (see Fig. 8). They are divided into two subsets, where one hundred configurations are assigned to be the training subset, and the other fifty configurations are assigned to be the validation subset. Corresponding nominal joint variables are calculated by nominal inverse kinematics (see Fig. 9). After normalization, they are used as input signals for the designed ANNs.

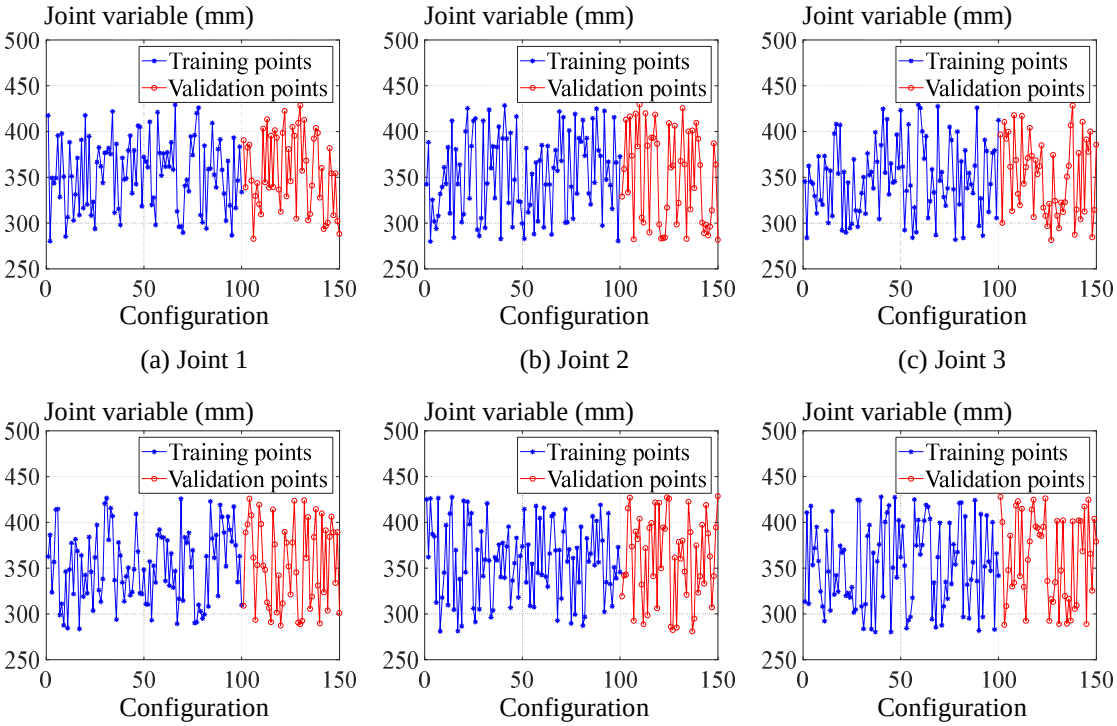

Fig. 9. Nominal joint variables of measuring configurations.

4.2. Experimental results

This subsection describes the main procedures and calibration results of the experiment. At a given measuring configuration, the actual position of P_j ($j=1\sim3$) is detected by the measuring arm three times, and the mean value is retained to inhibit the influence of measurement noise. Moreover, in order to eliminate the influence of tracking errors, we will not obtain the measurement information until the robot is stable for at least three seconds. Joint corrections can then be acquired based on the proposed method. The regularization parameter λ_0 is set to 0.05. The convergence threshold κ is set to $\sqrt{6}$ times the encoder resolution, which is 0.002mm. Examining the condition number of the matrix $\boldsymbol{PT}$ in all measuring configurations indicates that eighteen of them exceed the specified condition number $c=10^3$, and these configurations occur either at the boundary of the workspace or at large tilting angles. Coupled and decoupled networks can be trained after obtaining all pairs of nominal joint variables and corresponding joint corrections. The optimal number of hidden nodes is selected starting from a relatively small one.

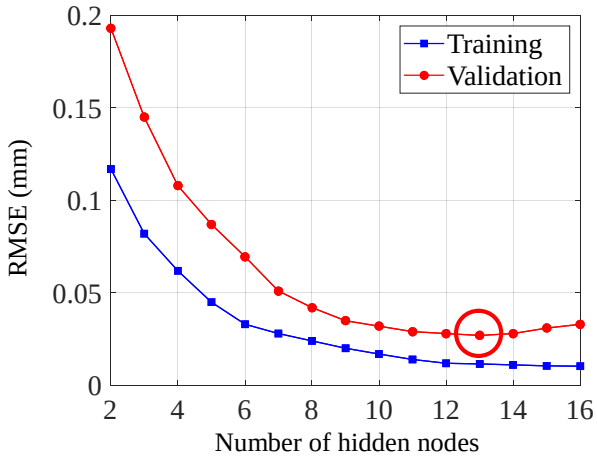

Fig. 10. Selection of the optimal number of hidden nodes for the coupled network.

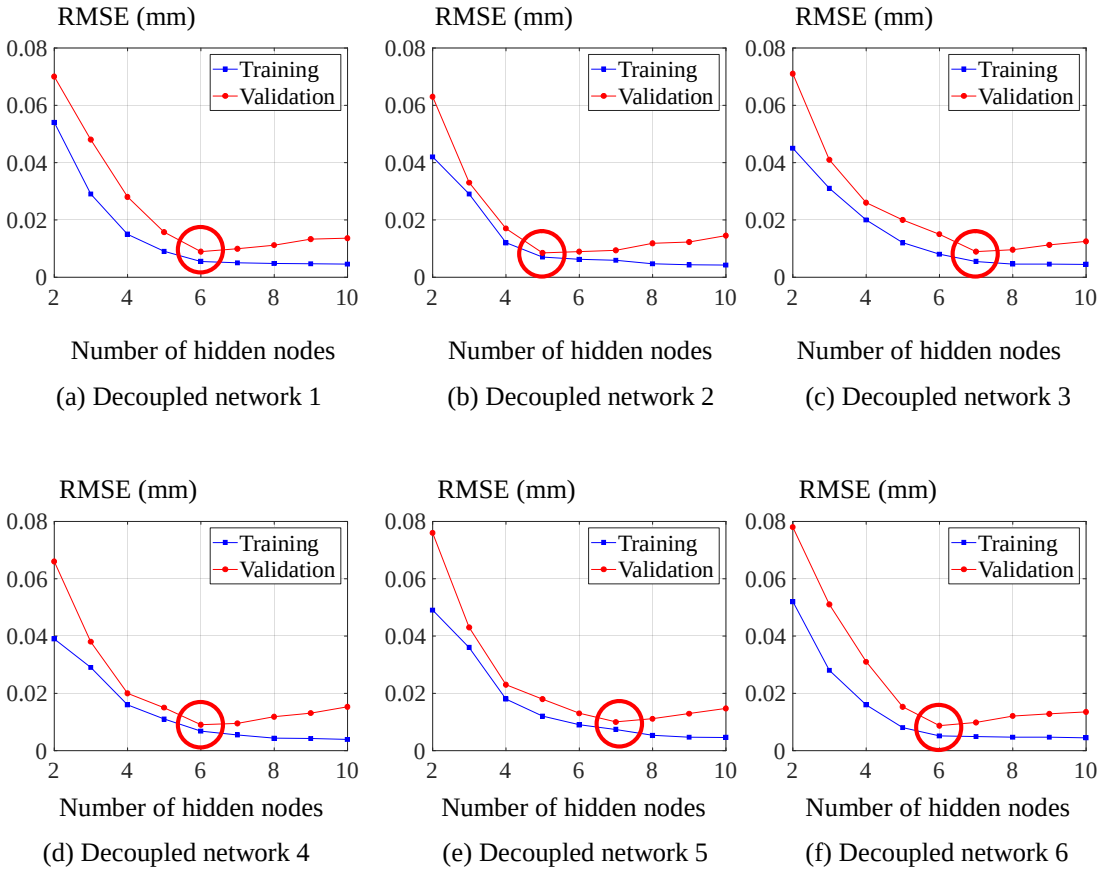

Fig. 11. Selection of the optimal number of hidden nodes for the decoupled networks.

It can be seen from Fig. 10 and Fig. 11 that the coupled and decoupled networks show a similar pattern. For instance, the RMSE on the validation set is always larger than that on the training set, which is expected because the validation set is not involved in the training process. Moreover, the RMSEs on both the training set and the validation set show a sharp drop with the initial increase of hidden nodes, which means more hidden nodes can improve the fitting capability in the initial stage. However, with the hidden nodes increase subsequently, the RMSE on the training set continues to decrease slowly, while that on the validation set increases due to overfitting. This indicates that the network has started to memorize the data of the training subset rather than learn the pattern of the function. Obviously, this is undesirable because the generalization capability has deteriorated. At a deeper level, a neural network with too many hidden nodes will unknowingly extract some of the residual variation (e.g., measurement noise and robot repeatability) as if that variation represents the intrinsic structure of the joint correction function. Therefore, the optimal number of hidden nodes can be determined at the inflection point of the performance in the validation set. According to this criterion, the optimal number of hidden nodes for the

coupled network is 13, whereas that for the decoupled network is about 6. To further compare these two architectures of networks, the average values of hidden nodes and training time for six decoupled networks are calculated, and a composite performance index of six decoupled networks is defined as

$$\text{Perf} = \sqrt{\sum_{i=1}^{6} \text{RMSE}_i^2} \tag{21}$$

where RMSE_i denotes the RMSE on the validation subset of the i th decoupled network. Note that performances on the validation subset are evaluated for comparison. As illustrated in Table 1, the performance of the decoupled networks is slightly better than that of the coupled network. However, the design efforts and total training time of the decoupled networks are more than those of the coupled network because we have to design and train six of the former independently.

It should be noted that we come to the opposite conclusion with [30], in which the coupled neural network fits better than the decoupled ones. However, this is explainable. The main reason is that the joint corrections do not have the coupling effect, and each joint can be controlled independently. For instance, joint motion error (joint correction) of joint 1 will not have any effects on that of joint 2. Another reason is that the coupled network may need more measuring configurations for training to achieve the same accuracy level as the decoupled networks. Calibration with different number of measuring configurations will be carried out to demonstrate this conjecture in the next subsection.

Table 1 Comparison between the coupled and decoupled networks.

Architecture	Average hidden nodes	Average training time (s)	Performance (mm)
Coupled network	13	29	0.027
Decoupled networks	6	8	0.022

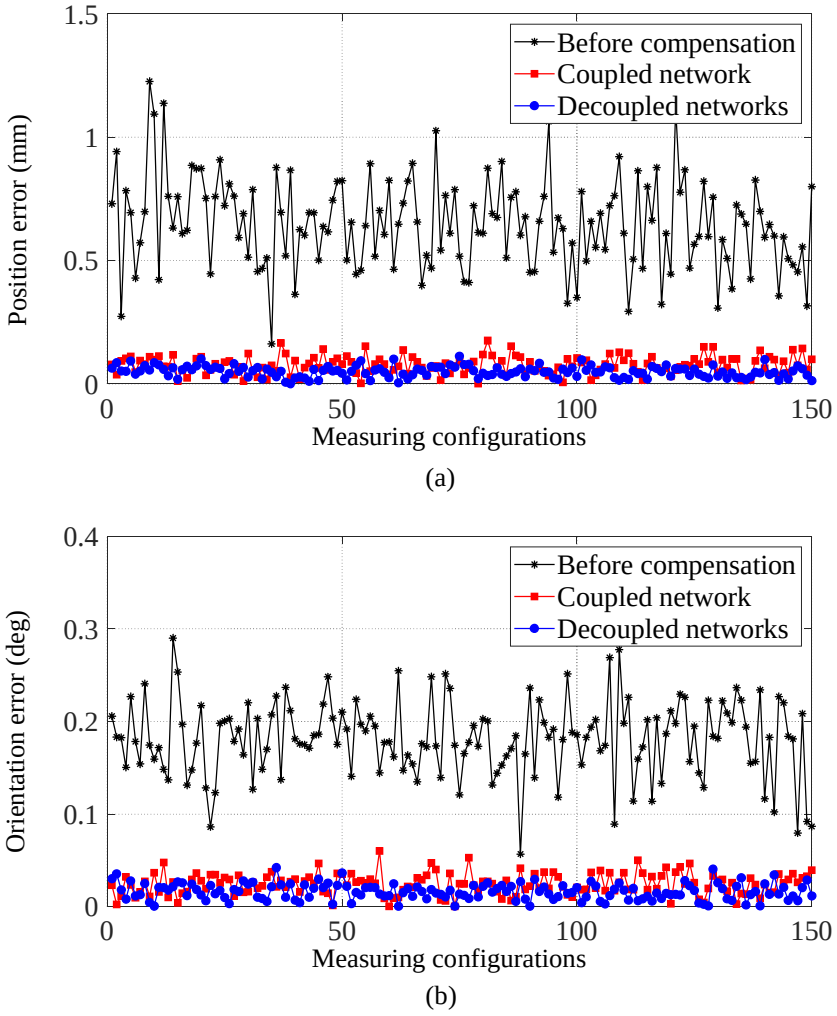

Fig. 12. Pose errors before and after compensation. (a) Position errors. (b) Orientation errors.

In order to demonstrate the calibration accuracy in the task workspace, error compensation is implemented using joint compensators based on two architectures of networks. Position error $\boldsymbol{\delta}_A$ and orientation error $\boldsymbol{\varepsilon}$ can be acquired by solving Eq. (4) with the measured position data in each configuration, and norms of them are calculated for evaluation. Fig. 12 shows pose errors before and after compensation in all measuring configurations, and Table 2 shows their statistical data. Mean values of the position/orientation errors are reduced by 88.16%/86.96% using the coupled network compared to those before compensation, whereas the use of the decoupled networks delivers 91.90%/90.22%. Maximum values of the position/orientation errors based on the coupled network can reach 0.175mm/0.060deg, whereas those based on the decoupled networks can reach 0.112mm/0.042deg. Both of them have

satisfactory pose error compensation results, which is expected since their capabilities have been confirmed in the joint space.

Table 2 Statistical data of pose errors before and after compensation.

Pose error		Before compensation	Coupled network	Decoupled networks
$\|\boldsymbol{\delta}_A\|$	MAX (mm)	1.225	0.175	0.112
	MEAN (mm)	0.642	0.076	0.052
	STD (mm)	0.163	0.041	0.024
$\|\boldsymbol{\varepsilon}\|$	MAX (deg)	0.290	0.060	0.042
	MEAN (deg)	0.184	0.024	0.018
	STD (deg)	0.041	0.013	0.009

4.3. Comparisons and discussions

For comparison, calibration experiments using the POE-based method [12] and the ordinary ANN-based joint compensator [22] are also carried out on the studied Stewart platform. The relationship between the calibration accuracy and the number of measuring configurations is also investigated.

In the comparative study, fifty, one hundred, one hundred and fifty measuring configurations are randomly selected to be the training subset, whereas the validation subset containing fifty configurations remains unchanged. Calibration results on the same validation subset using different methods are illustrated in Fig. 13. The mean values of position (orientation) errors are evaluated, and the standard deviations of them are included in the form of error bars. It can be seen from Fig. 13 that:

(1) The data from fifty configurations would be enough for the POE-based method since the calibration accuracy stays basically static with the increase of configurations. Although this model-based method requires a small number of configurations, the calibration accuracy is limited due to the unmodeled error sources.

(2) Ordinary compensator using the coupled network can improve the calibration accuracy continuously with more configurations. It outperforms the POE-based method when using more than one hundred and fifty configurations. However, since it does not consider near singularity problems, the

calculated joint corrections in some configurations are unstable, resulting in the data provided for network training with low confidence. Consequently, it is not as good as robust compensators.

(3) Similar to the ordinary compensator, the calibration accuracy of the proposed robust compensators is also closely related to the number of configurations, especially for the coupled network. Although the decoupled networks behave slightly better than the coupled network with a relatively small number of configurations (less than one hundred), they can have nearly the same calibration accuracy as long as we have enough configurations (more than one hundred and fifty). This result also proves the conjecture proposed in the last subsection.

(4) Both robust compensators outperform the others with more than one hundred configurations. When using one hundred and fifty configurations, the residual mean pose (position and orientation) errors of the proposed robust compensators are increased by about 50% compared with the POE-based method and about 40% compared with the ordinary compensator.

(5) The residual mean pose errors cannot reach zero but a relatively low bound. The final calibration accuracy is determined by three factors: the accuracy of the FARO measuring arm, the repeatability of the Stewart platform, and the residual errors of the trained neural network. The final calibration accuracy cannot be better than any of them.

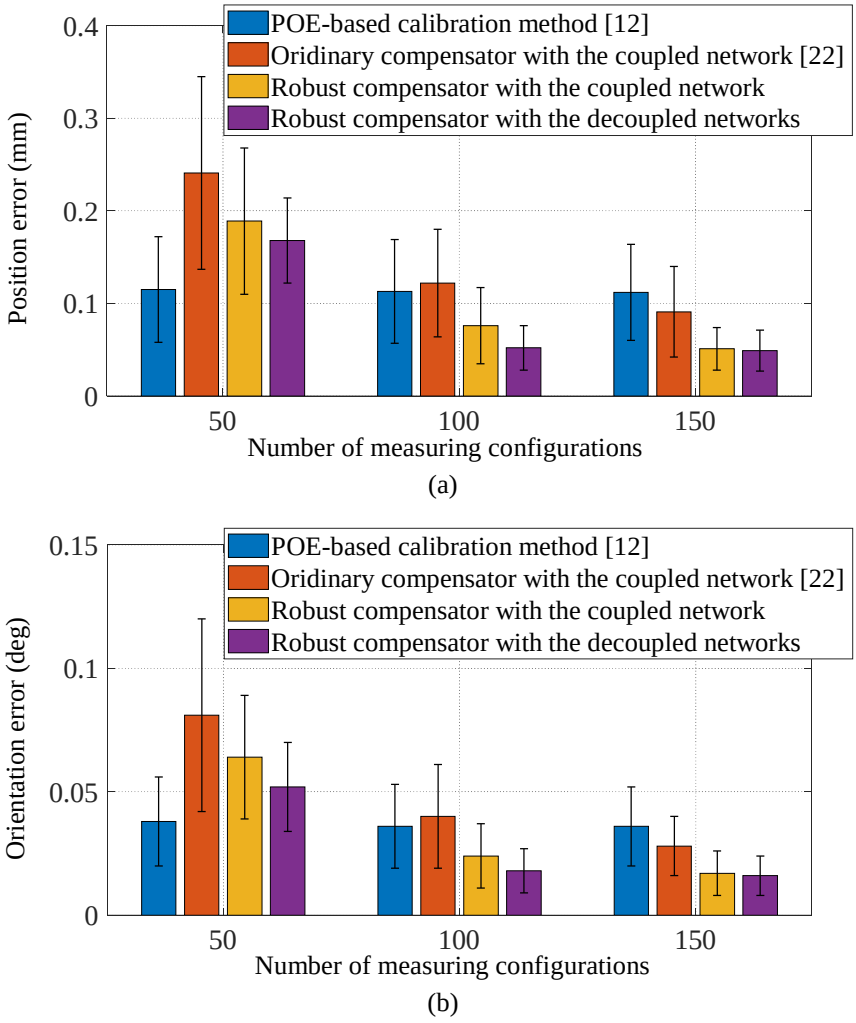

Fig. 13. Comparison of calibration methods with different number of measuring configurations. (a) Position errors. (b) Orientation errors.

5. Conclusions

This paper has proposed a novel calibration method by designing a robust joint compensator with ANNs to improve the static pose accuracy of a Stewart platform. The conclusions are drawn as follows:

(1) The pose error is treated as produced only by configuration-dependent joint motion errors equivalently, thus allowing the pose error to be eliminated by correcting the nominal joint variables directly. Based on this idea, joint corrections are acquired with point measurement considering near singular configurations, and the function between joint corrections and nominal joint variables is further approximated by the coupled/decoupled networks. Online real-time error compensation is implemented by designing a joint compensator embedded in the NC system.

(2) Experimental results on the Stewart platform prototype demonstrate the effectiveness of the proposed method. Robust compensators based on both coupled and decoupled networks have satisfactory calibration results. Although the decoupled networks behave slightly better than the coupled network when using a relatively small number of configurations, the former requires more design efforts and training time than the latter. They can achieve the same pose accuracy with about 0.05mm/0.02deg average position/orientation error as long as we have enough configurations.

(3) This method is general and systematic enough to be applied to calibrate other 6-DOF industrial robots, including both serial and parallel robots. However, it is worth investigating how to solve joint corrections for lower mobility robots and how to approximate the joint correction function better with other machine learning methods in further study.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China (grants 91948301 and 51721003), EU H2020-RISE-ECSASDP (grant 734272) and China Scholarship Council (No. 201906250074).

References

[1] Stewart D. A platform with six degrees of freedom. Proceedings of the Institution of Mechanical Engineers 1965;180:371–86. https://doi.org/10.1243/pime_proc_1965_180_029_02.

[2] Becerra-Vargas M, Belo EM. Application of H_∞ theory to a 6 DOF flight simulator motion base. Journal of the Brazilian Society of Mechanical Sciences and Engineering 2012;34:193–204. https://doi.org/10.1590/S1678-58782012000200011.

[3] Roth ZS, Mooring BW, Ravani B. An overview of robot calibration. IEEE Journal on Robotics and Automation 1987;3:377–85. https://doi.org/10.1109/JRA.1987.1087124.

[4] Karan B, Vukobratović M. Calibration and accuracy of manipulation robot models-An overview. Mechanism and Machine Theory 1994;29:479–500. https://doi.org/10.1016/0094-114x(94)90130-9.

[5] Li Z, Li S, Luo X. An overview of calibration technology of industrial robots. IEEE/CAA Journal of Automatica Sinica 2021;8:23–36. https://doi.org/10.1109/JAS.2020.1003381.

[6] Hayati S, Mirmirani M. Improving the absolute positioning accuracy of robot manipulators. Journal of Robotic Systems 1985;2:397–413. https://doi.org/10.1002/rob.4620020406.

[7] Stone HW, Sanderson AC, Neumann CF. Arm signature identification. IEEE International Conference on Robotics and Automation 1986;1:41–8. https://doi.org/10.1109/ROBOT.1986.1087664.

[8] Chen IM, Yang G, Tan CT, Yeo SH. Local POE model for robot kinematic calibration. Mechanism and Machine Theory 2001;36:1215–39. https://doi.org/10.1016/s0094-114x(01)00048-9.

[9] Li C, Wu Y, Lowe H, Li Z. POE-based robot kinematic calibration using axis configuration space and the adjoint error model. IEEE Transactions on Robotics 2016;32:1264–79. https://doi.org/10.1109/TRO.2016.2593042.

[10] Sharkawy AN, Aspragathos N. A comparative study of two methods for forward kinematics and Jacobian matrix determination. International Conference on Mechanical, System and Control Engineering 2017;179–83. https://doi.org/10.1109/ICMSC.2017.7959467.

[11] Zhuang H, Roth ZS. Method for kinematic calibration of stewart platforms. Journal of Robotic Systems 1993;10:391–405. https://doi.org/10.1002/rob.4620100306.

[12] Chen G, Kong L, Li Q, Wang H, Lin Z. Complete, minimal and continuous error models for the kinematic calibration of parallel manipulators based on POE formula. Mechanism and Machine Theory 2018;121:844–56. https://doi.org/10.1016/j.mechmachtheory.2017.11.003.

[13] Liu H, Huang T, Chetwynd DG. A general approach for geometric error modeling of lower mobility parallel manipulators. Journal of Mechanisms and Robotics 2011;3:021013. https://doi.org/10.1115/1.4003845.

[14] Mei B, Xie F, Liu XJ, Yang C. Elasto-geometrical error modeling and compensation of a five-axis parallel machining robot. Precision Engineering 2021;69:48–61. https://doi.org/10.1016/j.precisioneng.2021.01.007.

[15] Guo Y, Yin S, Ren Y, Zhu J, Yang S, Ye S. A multilevel calibration technique for an industrial robot with parallelogram mechanism. Precision Engineering 2015;40:261–72. https://doi.org/10.1016/j.precisioneng.2015.01.001.

[16] Chen D, Yuan P, Wang T, Cai Y, Xue L. A compensation method for enhancing aviation drilling robot accuracy based on co-kriging. International Journal of Precision Engineering and Manufacturing 2018;19:1133–42. https://doi.org/10.1007/s12541-018-0134-8.

[17] Cao S, Cheng Q, Guo Y, Zhu W, Wang H, Ke Y. Pose error compensation based on joint space division for 6-DOF robot manipulators. Precision Engineering 2022;74:195–204. https://doi.org/10.1016/j.precisioneng.2021.11.010.

[18] Bai Y. On the comparison of model-based and modeless robotic calibration based on a fuzzy interpolation method. International Journal of Advanced Manufacturing Technology 2007;31:1243–50. https://doi.org/10.1007/s00170-005-0278-4.

[19] Bai Y, Wang D. Calibrate parallel machine tools by using interval type-2 fuzzy interpolation method. International Journal of Advanced Manufacturing Technology 2017;93:3777–87. https://doi.org/10.1007/s00170-017-0781-4.

[20] Alici G, Shirinzadeh B. A systematic technique to estimate positioning errors for robot accuracy improvement using laser interferometry based sensing. Mechanism and Machine Theory 2005;40:879–906. https://doi.org/10.1016/j.mechmachtheory.2004.12.012.

[21] Bai M, Zhang M, Zhang H, Li M, Zhao J, Chen Z. Calibration method based on models and least-squares support vector regression enhancing robot position accuracy. IEEE Access 2021;9:136060–70. https://doi.org/10.1109/ACCESS.2021.3115949.

[22] Yu D, Cong D, Han J. Parallel robots pose accuracy compensation using artificial neural networks. International Conference on Machine Learning and Cybernetics 2005:3194–8. https://doi.org/10.1109/ICMLC.2005.1527493.

[23] Nguyen HN, Zhou J, Kang HJ. A calibration method for enhancing robot accuracy through integration of an extended Kalman filter algorithm and an artificial neural network. Neurocomputing 2015;151:996–1005. https://doi.org/10.1016/j.neucom.2014.03.085.

[24] Zhuang H, Roth ZS, Hamano F. Optimal design of robot accuracy compensators. IEEE Transactions on Robotics and Automation 1993;9:854–7. https://doi.org/10.1109/70.265930.

[25] Shamma JS, Whitney DE. A method for inverse robot calibration. Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME 1987;109:36–43. https://doi.org/10.1115/1.3143817.

[26] Marquardt DW. An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics 1963;11:431–41. https://doi.org/10.1137/0111030.

[27] Basheer IA, Hajmeer M. Artificial neural networks: fundamentals, computing, design, and application. Journal of Microbiological Methods 2000;43:3–31. https://doi.org/10.1016/s0167-7012(00)00201-3.

[28] Sharkawy AN. Principle of Neural Network and Its Main Types: Review. Journal of Advances in Applied & Computational Mathematics 2020;7:8–19. https://doi.org/10.15377/2409-5761.2020.07.2.

[29] International Organization for Standardization. Manipulating Industrial Robots—Performance Criteria and Related Test Methods. ISO 9283; 1998.

[30] Sharkawy AN, Koustoumpardis PN, Aspragathos N. Human–robot collisions detection for safe human–robot interaction using one multi-input–output neural network. Soft Computing 2020;24(9):6687–719. https://doi.org/10.1007/s00500-019-04306-7.