2016 IEEE International Conference on Robotics and Automation (ICRA)
Stockholm, Sweden, May 16-21, 2016

Calibration of Industry Robots with Consideration of Loading Effects Using Product-Of-Exponential (POE) and Gaussian Process (GP)

Wei Jing[1], Pey Yuen Tao[2], Guilin Yang[3] and Kenji Shimada[1]

Abstract— **Robot calibration is critical for industrial robot applications that require high accuracy. This paper presents a novel calibration method that utilizes Product-Of-Exponential (POE) and Gaussian Process (GP) regression to compensate for both geometric and non-geometric errors within the robot manipulator. Effects of a payload at the end-effector is also considered in the GP regression model in order to further improve robot positioning accuracy in the task space. Simulation and experimental results demonstrate the effectiveness of the proposed method. The experimental results show that the proposed method reduces norm pose error by 65.5% and 50.2% on average compared to conventional base-tool calibration and POE calibration respectively.**

I. INTRODUCTION

Robot serial manipulators are important tools for manufacturing automation in order to improve productivity and realize a healthy manufacturing sector. Many of these industrial applications require high accuracy. However, traditional industry robot manipulators usually have very good repeatability (e.g. usually upto sub mini-meter) but relatively poor task space accuracy. Traditionally, the robot manipulator is programmed using a teaching pendant and the robot is jogged to all the required way points. The presented method overcomes some accuracy issues and relies on good robot repeatability to perform the required tasks.

With the increasing need for more efficient robot programming in the high-mix-low-volume (HMLV) production commonly encountered by small-to-medium enterprises (SMEs) or the expected high product customization in Industry 4.0, robots have to be reprogrammed more frequently, and the traditional approach will not be sufficiently effective as it is a labor intensive, time consuming, highly skill-dependent process which causes downtime in the production lines. Other programming methods such as offline programming are required to improve the robot programming process. However, programming methods that rely on simulation or virtual representation of the robot manipulator will require high positioning accuracy to realize the paths programmed in the virtual environment.

[1]Wei Jing and Kenji Shimada is with Department of Mechanical Engineering, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA, 15213, USA. wj@andrew.cmu.edu, shimada@cmu.edu

[2]Pey Yuen Tao is with Singapore Institute of Manufacturing Technology, 71 Nanyang Drive, 638075 Singapore; and SIMTech-NUS Joint Lab (Industrial Robotics), c/o Department of Mechanical Engineering, National University of Singapore, 117575 Singapore. pytao@simtech.a-star.edu.sg

[3] Guilin Yang is with Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, Zhejiang 315201, China glyang@nimte.ac.cn

Accurately positioning the end-effector of the robot is a challenging task, given the various sources of errors. Thus, calibrating the robot to achieve good positioning accuracy is required to address these problems. Robot calibration methods have been studied intensively during the past few decades. In the past, researchers mainly adopted Denavit-Hartenberg (DH) parametrization of the robot kinematic model in the calibration process [1] [2] [3], which suffers from singularity problems when two adjacent axes are nearly parallel [4] and the inability to model small arbitrary misalignments of adjacent frames [5]. Park [6] proposed to use the Product-Of-Exponential (POE) model for robotic calibration to better model the misalignments of adjacent frames. Subsequently, Okamura and Park [7] extended the work and proposed an iterative least square optimization method for the calibration process. After that, Chen et. al. [4] proposed a Local POE method that led to better accuracy and lower computational cost.

Though efforts have been made to reduce geometric error in kinematics, it is worth noting that there are also non-geometric error sources [8] such as link deflection and joint compliance, which affects the robot accuracy as well. Jang et. al. [9] combined the DH model with Neural Network (NN) to compensate for both geometric and non-geometric errors. Subsequently, Aoyagi et. al. [10] utilized Genetic Algorithm (GA) to select optimal calibration poses besides applying the DH model and NN to compensate for both types of errors. Tao and Yang [11] also addressed the non-geometric errors by compensating for them using NN on top of local POE model calibration, which effectively reduces the pose error when compared to using POE calibration alone. However, NN still suffers from high computational cost and may suffer from accuracy problems when the population of training samples is small.

In this paper, a novel calibration method is proposed where the POE model is used to model the robot kinematics, and the parameters within the model are calibrated based on measurement data. Gaussian Process (GP) regression is used to compensate for the residual errors. Joint deflections due to the payload are considered in the GP model to further reduce the pose error. The GP used in this paper is based on Bayesian inference and is suitable for regression problems with a small number of feature sizes and a small amount of training data, while NNs usually work well when the training data is large. For robot calibration application, usually a few hundred measurement data points will be collected as training data. Thus, GP is expected to work better than NN for this application with small amount of training data.

The main contributions of this paper are listed as follows:

- The proposed method combines POE and GP to compensate for both geometric and non-geometric errors,
- The proposed method considers the payload of end-effector in the GP model, which further improves the calibration accuracy, and
- The proposed method is verified through simulations and experimental studies.

II. PROBLEM FORMULATION

The goal of robot calibration is to identify the robot model that minimizes the difference between the computed and measured end-effector pose. Given a robot with n joints and the kinematic model f, the theoretical or computed pose $\boldsymbol{T}$ of end-effector is given by:

$$\boldsymbol{T} = f(\boldsymbol{q}), \quad \boldsymbol{T} \in SE(3),$$

where $\boldsymbol{q} \in R^{n\times 1}$ is the joint angles; n is the number of joints of the robot; $SE(3)$ stands for *Special Euclidean Group*.

However, in reality, the true kinematic model is slightly different from the idealized model, which causes pose error of the end-effector. Because such pose error affects the robot accuracy, calibration is required to modify the robot kinematic model to minimize the pose error of the end-effector.

The calibration is to find an accurate kinematic model f such that the difference between $f(\boldsymbol{q})$ and measurement $\boldsymbol{T}'$ is minimized:

$$\begin{aligned} &\min \sum \|\boldsymbol{T}'\boldsymbol{T}^{-1} - \boldsymbol{I}\| \\ =&\min_{f} \sum \|\boldsymbol{T}' f(\boldsymbol{q})^{-1} - \boldsymbol{I}\|, \end{aligned}$$

where $\boldsymbol{I} \in R^{4\times 4}$ is the identity matrix.

In this paper, GP is used to compensate for the residual errors and also to consider the effects of a payload at the end-effector. Thus the new overall kinematic model $\bar{f}$ can be written as $\bar{f}(\boldsymbol{q}, m) = G(\boldsymbol{q}, m) f(\boldsymbol{q})$ by appending GP in addition to the previous kinematic model f. Then the formulation used in this paper is shown as follows:

$$\begin{aligned} &\min \sum \|\boldsymbol{T}'\boldsymbol{T}^{-1} - \boldsymbol{I}\| \\ =&\min_{\bar{f}} \sum \|\boldsymbol{T}' \bar{f}(\boldsymbol{q}, m)^{-1} - \boldsymbol{I}\| \\ =&\min_{G,f} \sum \|\boldsymbol{T}' f(\boldsymbol{q})^{-1} G(\boldsymbol{q}, m)^{-1} - \boldsymbol{I}\|, \end{aligned}$$

where m is the payload of the end-effector, and $G(\boldsymbol{q}, m) = \boldsymbol{T}_g \in SE(3)$ is the homogeneous transformation representation of the GP model ($g(\boldsymbol{q}, m) \in R^{6\times 1}$). In the proposed method, both the POE model and GP model are calibrated to minimize the errors between the measured and computed end-effector poses, the details of POE model and GP model is explained in Section III.

III. METHODOLOGY

This paper presents a novel two-step calibration method that minimizes the robotic kinematic error. In the first step, the proposed method utilizes a POE model-based calibration to identify a set of model parameters that minimizes the errors between the computed and measured end-effector poses. Subsequently, GP is used to compensate for residual errors taking into account of payload at the end-effector.

A. Step 1: POE Model Calibration

1) POE Model: The POE calibration method is adopted from previous work by Chen et. al. [4]. Unlike the DH model, the POE model effectively models transformation between arbitrary frames, which makes it suitable to eliminate arbitrary misalignments in adjacent frames during calibration.

In the POE model, ${}^{i}\boldsymbol{T}_{i-1} \in SE(3)$ is a homogeneous transformation matrix that models the geometric relationship between adjacent frame i and $i-1$. Thus, the POE representation of the transformation matrix between two adjacent frames with a joint rotation of q_i is defined in the following form:

$$ {}^{i}\boldsymbol{T}_{i-1}(q_i) = {}^{i}\boldsymbol{T}_{i-1}(0) e^{\boldsymbol{s}_i q_i}, \tag{1}$$

where $\boldsymbol{s}_i \in se(3)$ is the twist that models the rotational/prismatic joint axis and ${}^{i}\boldsymbol{T}_{i-1}(0) \in SE(3)$ is the initial pose of the frame i, which could also be represented as $e^{\hat{\boldsymbol{p}}_i} ={}^{i} \boldsymbol{T}_{i-1}(0), \hat{\boldsymbol{p}}_i \in se(3)$. Thus, Eq. (1) can be rewritten as:

$$ {}^{i}\boldsymbol{T}_{i-1}(q_i) = e^{\hat{\boldsymbol{p}}_i} e^{\boldsymbol{s}_i q_i}. \tag{2}$$

The transformation matrix describing the relative pose of frame $n+1$ in frame 0 of a n-jointed robot is given by:

$$ {}^{0}\boldsymbol{T}_n = \prod_{i=1}^{n} ({}^{i}\boldsymbol{T}_{i-1}(0) e^{\boldsymbol{s}_i q_i}) {}^{n}\boldsymbol{T}_{n+1}(0) = \prod_{i=1}^{n} (e^{\hat{\boldsymbol{p}}_i} e^{\boldsymbol{s}_i q_i}) e^{\hat{\boldsymbol{p}}_{n+1}}, \tag{3}$$

where $\prod$ is defined as a non-commutative product of matrices:

$$\prod_{i=1}^{k} \boldsymbol{X}_i = \boldsymbol{X}_1 \boldsymbol{X}_2 ... \boldsymbol{X}_k$$

2) POE Model Calibration: The goal of POE model calibration is to identify a set of parameters $\boldsymbol{P} = [\boldsymbol{p}_1^T, \boldsymbol{p}_2^T, ..., \boldsymbol{p}_{n+1}^T]^T \in R^{6(n+1)\times 1}$ such that the error between the computed and measured end-effector poses is minimized. Note that $e^{\hat{\boldsymbol{p}}_i} = {}^{i}\boldsymbol{T}_{i-1}(0)$ models the zero pose transformation matrix of frame i to $i-1$ when $q_i = 0$; $\boldsymbol{p}_i \in R^{6\times 1}$ is the vector representation of $\hat{\boldsymbol{p}}_i$.

Then the POE model calibration in the first step is formulated as a linear model $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{x}$ by taking the difference between the measured pose $\boldsymbol{T}'$ and the computed pose $\boldsymbol{T}$, where

$$\begin{aligned} \boldsymbol{Y} &= [\boldsymbol{y}_1^T, \boldsymbol{y}_2^T, \cdots, \boldsymbol{y}_k^T]^T \in R^{6k\times 1} \\ \boldsymbol{A} &= \begin{bmatrix} \boldsymbol{A}_1 \\ \vdots \\ \boldsymbol{A}_k \end{bmatrix} \in R^{6k\times 6(n+1)} \\ \boldsymbol{x} &= [\delta\boldsymbol{p}_1^T, \delta\boldsymbol{p}_2^T, ..., \delta\boldsymbol{p}_{n+1}^T]^T \in R^{6(n+1)\times 1}, \end{aligned}$$

where k is the number of measured poses and corresponding joint angles used for calibration, n is the number of joints in the robot, and

$$\begin{aligned}
&\boldsymbol{y}_i = log({}^0\boldsymbol{T}_n' \cdot {}^0\boldsymbol{T}_n^{-1})^V \quad \boldsymbol{y}_i \in R^{6\times 1}\\
&\boldsymbol{A}_i = [\boldsymbol{Ad_0}, \boldsymbol{Ad_1}, ..., \boldsymbol{Ad_{n+1}}] \in R^{6\times 6(n+1)}\\
&\boldsymbol{y}_i = \boldsymbol{A}_i\boldsymbol{x},
\end{aligned}$$

where ${}^0\boldsymbol{T}_n'$ is the measured end-effector pose, and ${}^0\boldsymbol{T}_n$ is the computed pose using the robot kinematic model; $\log(\boldsymbol{T})^V$ converts the homogeneous transformation matrix $\boldsymbol{T} \in SE(3)$ to a $R^{6\times 1}$ vector representation of the position and orientation [4]; $\boldsymbol{y}_i$ indicates the difference between the measured and computed poses; $\boldsymbol{x}$ is the update to the model parameters to minimize $\boldsymbol{y}_i$; $\boldsymbol{A_m}$ is the matrix mapping the model parameter errors to the end-effector errors; and, $\boldsymbol{Ad_i}$ is the adjoint representation of ${}^i\boldsymbol{T}_{i+1}$ [6] [4].

Let $\boldsymbol{A}^*$ be the pseudoinverse of $\boldsymbol{A}$, then $\boldsymbol{x}$ can be obtained by using $\boldsymbol{x} = \boldsymbol{A}^*\boldsymbol{Y}$. A new set of model parameters $\boldsymbol{p}$ can be obtained by updating $\boldsymbol{p}$ with $\boldsymbol{p} = \boldsymbol{p} + \boldsymbol{\delta p}$. This update is done iteratively until the errors $\boldsymbol{y}_i$ converge or a pre-defined number of iterations has passed.

B. Step 2: GP Regression for Residual Error

1) Gaussian Process: Gaussian Process [12] is a commonly used machine learning method for regression problems. It has been used in robotic kinematics modelling in previous work [13]. GP models the observations as multivariate Gaussian distribution, thus the output of any new input could be predicted through Bayesian inference.

Given $(\boldsymbol{x_i}, y_i)$ as a pair of input and output, where $\boldsymbol{x_i} \in R^{N\times 1}, y_i \in R$, then the multivariate Gaussian distribution model could be written as:

$$\boldsymbol{y} \sim N(\boldsymbol{0}, \boldsymbol{K}) \tag{4}$$

where $\boldsymbol{K}$ is the Gram matrix [14] that models the covariance between the inputs. In this paper, the quadratic form [14] is used to determine $\boldsymbol{K}$:

$$K_{ij} = \theta_0 \exp(-\frac{\theta_1(\boldsymbol{x_i} - \boldsymbol{x_j})^2}{2}) + \theta_2 + \theta_3\boldsymbol{x_i^T}\boldsymbol{x_j} \tag{5}$$

where θ_i for $i = 0, \cdots, 3$ are the tunable variables.

Therefore, given the observations $(\boldsymbol{x}, \boldsymbol{y})$, if a new input $\boldsymbol{x'}$ is presented, the new distribution becomes:

$$\begin{bmatrix}\boldsymbol{y}\\ y'\end{bmatrix} \sim N(\boldsymbol{0}, \begin{bmatrix}\boldsymbol{K} & \boldsymbol{K_1}\\ \boldsymbol{K_1^T} & \boldsymbol{K_2}\end{bmatrix}). \tag{6}$$

Then the conditional probabilistic distribution of y' associated with new input $\boldsymbol{x}'$ is also given as a Gaussian distribution:

$$y'|\boldsymbol{y} \sim N(\boldsymbol{K_1}\boldsymbol{K^{-1}}\boldsymbol{y}, \boldsymbol{K_2} - \boldsymbol{K_1}\boldsymbol{K^{-1}}\boldsymbol{K_1^T}). \tag{7}$$

According to Eq. (7), the output of future input can be predicted based on known observations. Note that the computational complexity of GP is close to $O(n^3)$, where n is the number of training data points, due to the inversion of the matrix, which makes GP less scalable. However, for a small amount of training data (e.g. a few hundred training data), the computational speed is acceptable.

2) GP for Error Compensation: The effect of the load (including the weight of the arm and the weight of the end-effector) on positioning errors of the end-effector can be formulated as a function of the joint angles and the weight of the end-effector, which can always be lumped into a 4×4 transformation matrix, such that $\boldsymbol{T}_{gp} = G(\boldsymbol{q}, m) \in SE(3)$).

After the POE calibration in the first step to identify the best set of model parameters to minimize the error between the measured and computed end-effector poses, there are still residual errors from non-geometric error sources that cannot be fully compensated by only changing the model parameters [15]. Let $\boldsymbol{p'}$ be the calibrated initial pose from the POE calibration, the remaining residual error is written as the difference of measured pose $\boldsymbol{T'}$ and the pose computed from calibrated POE model:

$$\boldsymbol{\epsilon} = \log(\boldsymbol{T}'(\prod_{i=1}^{n}(e^{\hat{\boldsymbol{p}}_i{}'}e^{\boldsymbol{s}_i q_i})e^{\hat{\boldsymbol{p}}'_{\boldsymbol{n+1}}})^{-1})^V, \tag{8}$$

and the overall kinematic equation with GP compensation is:

$${}^0\boldsymbol{T}_n = \boldsymbol{T}_{gp}(\prod_{i=1}^{n}(e^{\hat{\boldsymbol{p}}_i{}'}e^{s_i q_i})e^{\hat{\boldsymbol{p}}'_{\boldsymbol{n+1}}}). \tag{9}$$

Therefore, in the second step, the minimization problem that minimizes the overall error with GP compensation is formulated as:

$$\min_G \|\log(\boldsymbol{T}'(\prod_{i=1}^{n}(e^{\hat{\boldsymbol{p}}_i{}'}e^{\boldsymbol{s}_i q_i})e^{\hat{\boldsymbol{p}}'_{\boldsymbol{n+1}}})^{-1}G(\boldsymbol{q}, m)^{-1})^V\|, \tag{10}$$

where $\boldsymbol{T'}$ is the measured pose of the end-effector; $\log\boldsymbol{T}^V$ converts the 4×4 homogeneous transformation matrix to 6×1 vector; $g(\boldsymbol{q}, m) \in R^{6\times 1}$ is the GP model; and, $G(\boldsymbol{q}, m) \in SE(3)$ is the matrix exponential of $g(\boldsymbol{q}, m)$. As $G(\boldsymbol{q}, m)$ is an element of $SE(3)$, there exists at least one $\hat{g}(\boldsymbol{q}, m) \in se(3)$ such that $e^{\hat{g}(\boldsymbol{q},m)} = G(q, m)$, in which $\hat{g}(\boldsymbol{q}, m) \in se(3)$ is always associated with a unique vector $g(\boldsymbol{q}, m) \in R^{6\times 1}$. The minimization problem formulated in Eq. (10) is then solved by computing the GP model.

Therefore, the residual pose error is modelled by GP such that given the error $\boldsymbol{\epsilon}$ in Eq. (8) between the measured and computed poses, as well as the inputs $\boldsymbol{x} = [q_1, q_2, ...q_n, m]^T$, if a new input $\boldsymbol{x'}$ is presented, the error ϵ_i' $(i = 1, 2, \cdots, 6)$ in each axis can be predicted by the following Gaussian distribution:

$$\epsilon_i'|\boldsymbol{\epsilon} \sim N(\boldsymbol{K_1}\boldsymbol{K^{-1}}\boldsymbol{\epsilon}, \boldsymbol{K_2} - \boldsymbol{K_1}\boldsymbol{K^{-1}}\boldsymbol{K_1^T}) \quad i = 1, 2, \cdots, 6 \tag{11}$$

and the input to the GP is the joint angles and payload on the end-effector:

$$\boldsymbol{x} = [q_1, q_2, ...q_n, m]^T, \tag{12}$$

where q_i is the joint angle of i^{th} joint. Therefore, the minimization problem in Eq. (10) is then solved by finding the GP model that models the mapping from $\boldsymbol{x}$ to residual error $\boldsymbol{\epsilon}$.

Then the compensated $\boldsymbol{y'} \in R^{6\times 1}$ can be written as:

$$\boldsymbol{y'} = \log(\boldsymbol{T}_{gp}(\prod_{i=1}^{n}(e^{\hat{\boldsymbol{p}}_i{}'}e^{s_i q_i})e^{\hat{\boldsymbol{p}}'_{\boldsymbol{n+1}}}))^V \tag{13}$$

where $\boldsymbol{T}_{gp} = G(\boldsymbol{q}, m), \boldsymbol{T}_{gp} \in SE(3)$ is the compensation of error by GP model. As a machine learning model for compensation, $\boldsymbol{T}_{gp}$ could be put either before of after the POE model; in this paper, it is put before the POE model in order to save computational cost.

IV. RESULTS AND DISCUSSION

In this section, the effectiveness of the proposed method is verified through simulation and experimental results. The proposed method is shown to significantly improve robot accuracy compared to the conventional base-tool calibration, POE model-based calibration and the POE+GP calibration without considering the payload at the end-effector in the GP model.

Different calibration methods are first compared using simulated data, followed by the same study using experimental data. The base-tool calibration method is adopted from previous work [11] in order to identify the relative pose of the robot base frame in the measurement frame and the relative pose of the end-effector frame in the frame attached to the last link of the robot. The GP implementation is based on open source project scikit-learn [16] with quadratic covariance function. According to Eq. (10), (11) and (12), the training of GP model is conducted by using $\boldsymbol{x}$ in Eq. (12) as input of GP model and $\boldsymbol{\epsilon}$ in Eq. (11) as output of GP model.

A. Simulation Results

In the simulation study conducted using Matlab and Python, the ABB IRB 4400 robot model is used with slightly perturbed kinematic parameters. The loading effect is simulated by adding different payloads on the end-effector, and the loading model used in this simulation is adopted from previous work [15]. Joint data are randomly generated within the workspace, a few loading weights (1kg, 2.5kg, 3.5kg, 5kg and 7.5kg) are included and randomly combined with the joint data.

A total of 300 training data points were randomly generated within the workspace, and a 10-fold cross validation was performed. The results shown in Figure. 1 indicate that the proposed method has the highest calibration accuracy compared with the other three calibration methods on a 10-fold cross validation of the training data.

Subsequently, all 300 training data points were used to obtain the calibrated model of the robot, and the updated model was tested using three different testing data sets, each consisting of 100 testing data. As shown in Table I, POE+GP calibration considering the robot payload yields the highest calibration accuracy among the four calibration methods using the simulated testing data. The proposed method reduces the norm pose error by 87.5%, 71.3%, 63.1% on average compared to base-tool calibration, POE calibration and POE + GP calibration without considering the payload respectively.

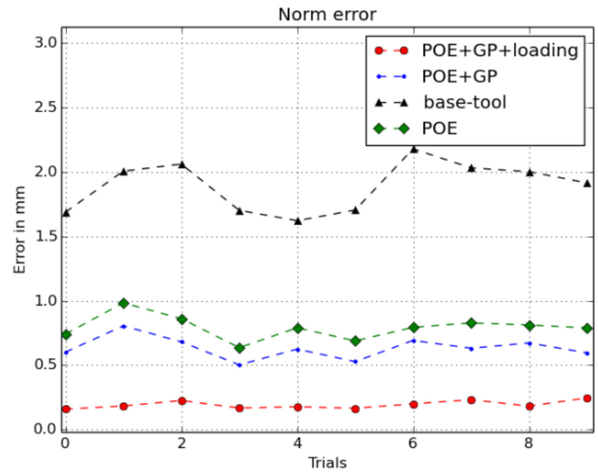

Fig. 1: 10-fold cross validation of training data, comparing all methods with simulation data

TABLE I: Simulation Results on Test Dataset

	Mean Error of Dataset (mm) 1	2	3
Base-tool Calibration	1.94	1.93	1.89
POE Calibration	0.90	0.81	0.81
POE + GP	0.71	0.63	0.62
POE + GP with Loading	0.23	0.24	0.25

B. Experiment Results

The experiment was carried out using an ABB IRB-4400 robot. The position and orientation of the robot end-effector was measured using Leica Absolute Tracker AT901-MR with a T-MAC 6 Degree-of-Freedom (DOF) sensor, which has a measurement accuracy of $30\mu m$. A few hundred measurements were collected randomly within the workspace; 5 mechanical tools with different weights $(1.46kg, 3.94kg, 4.77kg, 6.26kg, 9.72kg)$ were used in the experiment. The experimental setup is shown in Figure. 2.

The experiment was divided into two parts. In the first part, the experiment was performed with the T-MAC 6 DOF sensor on the end-effector without any additional loads, in order to demonstrate the effectiveness of using GP to compensate the residual error. In the second part, measurement data was collected by attaching different payloads at the end-effector. After randomly dividing the measurement data into training and testing data sets, a 10-fold cross validation was performed on the training data and finally the calibrated models, obtained with the four calibration methods using the training dataset, were tested using the testing data. The details of the experiments are shown in Sections IV-B.1 and IV-B.2.

1) Experiment with the Same Payload: In the first part of the experiment, POE calibration with GP compensation was adopted using the training data set obtained through experiments using the same payload at the end-effector. In this part, only the T-MAC 6 DOF sensor was mounted on the end-

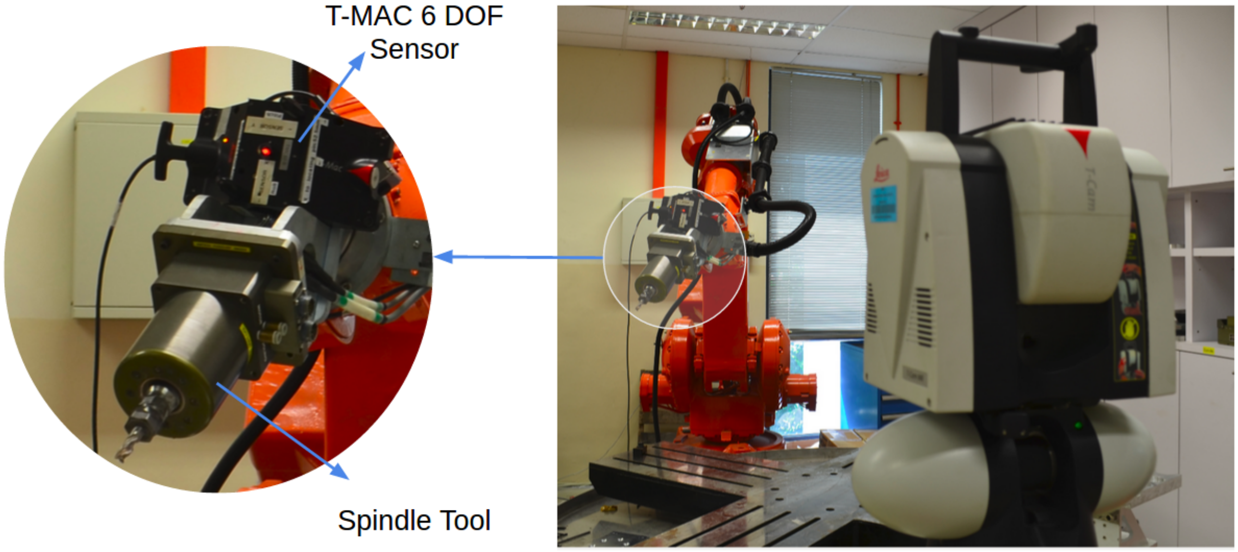

Fig. 2: The experiment setup

effector, which has a total weight of $1.49kg$. As the payload was the same, there was no difference between the POE+GP with and without considering the payload. Thus, the 10-fold cross validation was only performed among 3 calibration methods using the training data set. The calibration result is shown in Figure. 3, where the mean norm errors of tool base calibration, POE calibration and POE + GP calibration are $1.10mm$, $0.80mm$ and $0.40mm$, respectively.

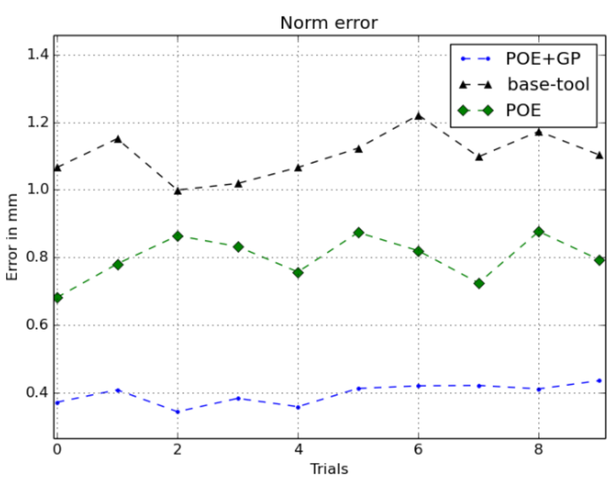

Fig. 3: 10-fold cross validation of training data set, comparing three methods with experimental data with the same loading

2) Experiment with Different Payload: In the second part of the experiment, the experiment data set of 5 mechanical tools with different weights $(1.46kg, 3.94kg, 4.77kg, 6.26kg, 9.72kg)$ was used. After collecting the data, the data is randomly divided into a training data set and three testing data sets. A 10-fold cross validation on the 300 training measurements was performed, and the calibrated models were obtained using the 4 calibration methods and the training data. The results illustrated in Figure. 4 demonstrate the effectiveness of the proposed method.

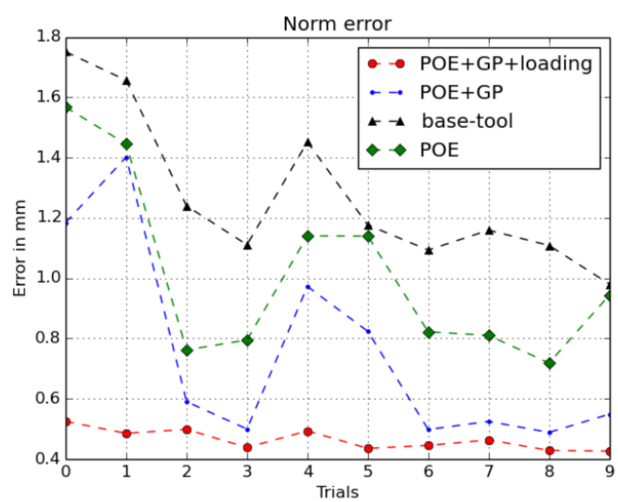

Fig. 4: 10-fold cross validation of training data set, comparing all methods with experimental data

To further validate the proposed method, the calibrated models were tested using 3 different testing data sets, each consisting of 100 measurements, which have not been used to train the models. As shown in Table II, POE+GP calibration considering payload yields the highest accuracy among the four calibration methods. The proposed method reduces the norm pose error by 65.5%, 50.2%, 48.2% on average compared to base-tool calibration, POE calibration and POE + GP calibration without considering payload respectively.

TABLE II: Experiment Results on Test Dataset

	Mean Error of Dataset (mm) 1	2	3
Base-tool Calibration	1.17	1.34	1.76
POE Calibration	0.87	0.82	1.30
POE + GP	0.83	0.78	1.28
POE + GP with Loading	0.43	0.46	0.57

C. Discussion

The results presented in this section show that the proposed method of POE+GP with consideration of loading effects yields the least error compared to existing methods. This is possibly due to a number of factors. Firstly, GP regression is suitable for small amount of training data set because of its robustness and smoothness, and it is also considered as a non-parametric model which should work well in many situations with little effort on tuning GP model parameters. One well-known limitation of GP is its poor scalability, but in this problem it is not a major issue as the size of the training data set is relatively small (e.g. a few hundred training data). In addition, the experimental results confirmed that the proposed new scheme considering the payload at the end-effector improves the modelling accuracy of the GP.

V. CONCLUSIONS

In this paper, a novel two-step robot calibration method is presented that combines POE and GP to compensate for both geometric and non-geometric error sources in an industrial robot. By considering the effects of the payload at the end-effector on the robot as a feature of the GP model, it is demonstrated that the proposed method of POE+GP with consideration of loading effects yields the smallest calibration error compared to previous methods in both computational simulation and physical experiments. For future work, it is possible to use only a localized GP model for the error compensation in order to improve the computational efficiency. Alternatively, it is also possible to consider using GP to compensate for the inverse kinematics such that the calibration results can be easily applied in an industrial robot.

ACKNOWLEDGMENT

The authors acknowledge the support from the Collaborative Research Project under the SIMTech-NUS Joint Laboratory (Industrial Robotics), Ref: U13-R-024AU and support from the SERC Industrial Robotics Program Work Package 7, project number: 1225100007.

REFERENCES

[1] S. A. Hayati, "Robot arm geometric link parameter estimation," in *IEEE Conference on Decision and Control*. IEEE, 1983, pp. 1477–1483.

[2] D. Whitney, C. Lozinski, and J. M. Rourke, "Industrial robot forward calibration method and results," *Journal of dynamic systems, measurement, and control*, vol. 108, no. 1, pp. 1–8, 1986.

[3] S. Hayati, K. Tso, and G. Roston, "Robot geometry calibration," in *IEEE International Conference on Robotics and Automation*. IEEE, 1988, pp. 947–951.

[4] 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, pp. 1215–1239, 2001.

[5] I.-M. Chen and G. Yang, "Kinematic calibration of modular reconfigurable robots using product-of-exponentials formula," *Journal of Robotic Systems*, vol. 14, no. 11, pp. 807–821, 1997.

[6] F. C. Park, "Computational aspects of the product-of-exponentials formula for robot kinematics," *IEEE Transactions on Automatic Control*, vol. 39, no. 3, pp. 643–647, 1994.

[7] K. Okamura and F. Park, "Kinematic calibration using the product of exponentials formula," *Robotica*, vol. 14, no. 04, pp. 415–421, 1996.

[8] B. Greenway, "Robot accuracy," *Industrial Robot: An International Journal*, vol. 27, no. 4, pp. 257–265, 2000.

[9] J. H. Jang, S. H. Kim, and Y. K. Kwak, "Calibration of geometric and non-geometric errors of an industrial robot," *Robotica*, vol. 19, no. 03, pp. 311–321, 2001.

[10] S. Aoyagi, A. Kohama, Y. Nakata, Y. Hayano, and M. Suzuki, "Improvement of robot accuracy by calibrating kinematic model using a laser tracking system-compensation of non-geometric errors using neural networks and selection of optimal measuring points using genetic algorithm," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 5660–5665.

[11] P. Tao and G. Yang, "Calibration of industrial robots with product-of-exponential (poe) model and adaptive neural networks," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015, pp. 1448–1454.

[12] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: the MIT Press, 2006.

[13] S. Park, S. K. Mustafa, and K. Shimada, "Learning-based robot control with localized sparse online Gaussian process," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1202–1207.

[14] C. M. Bishop, *Pattern Recognition and Machine Learning*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[15] P. Y. Tao, G. Yang, and M. Tomizuka, "A calibration framework for industrial robotic work cells," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, 2013, pp. 1637–1642.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.