applied sciences

MDPI

Article

A Novel Indirect Calibration Approach for Robot Positioning Error Compensation Based on Neural Network and Hand-Eye Vision

Chi-Tho Cao [1], Van-Phu Do [1,2] and Byung-Ryong Lee [1,2,*]

[1] School of Mechanic and Automotive Engineering, University of Ulsan, Daehak-ro 93, Nam-gu, Ulsan 44610, Korea; thocao92@gmail.com (C.-T.C.); abeobk@gmail.com (V.-P.D.)
[2] Abeosystem Co, LTD, Daehak-ro 93, Nam-gu, Ulsan 44610, Korea
* Correspondence: brlee@ulsan.ac.kr; Tel.: +82-010-2558-2861; Fax: +82-052-259-1680

Received: 16 April 2019; Accepted: 9 May 2019; Published: 11 May 2019

Featured Application: This study aims to improve the absolute position error of robot manipulators for vehicle assembly line without using an expensive external apparatus.

Abstract: It is well known that most of the industrial robots have excellent repeatability in positioning. However, the absolute position errors of industrial robots are relatively poor, and in some cases the error may reach even several millimeters, which make it difficult to apply the robot system to vehicle assembly lines that need small position errors. In this paper, we have studied a method to reduce the absolute position error of robots using machine vision and neural network. The position/orientation of robot tool-end is compensated using a vision-based approach combined with a neural network, where a novel indirect calibration approach is presented in order to gather information for training the neural network. In the simulation, the proposed compensation algorithm was found to reduce the positional error to 98%. On average, the absolute position error was 0.029 mm. The application of the proposed algorithm in the actual robot experiment reduced the error to 50.3%, averaging 1.79 mm.

Keywords: error compensation; hand-eye calibration; absolute accuracy; neural network

1. Introduction

The repeated and monotonous manual work is continuously reintegrated by flexible manufacturing systems. To build a flexible manufacturing system, adopting an intelligent robot system is essential to distinguish workpieces within a workspace, perceive a situation and manipulate themselves autonomously [1]. Among the intelligent robot systems, vision-based robot systems have been developed continuously to improve the quality and efficiency of the manufacturing system such as arc welding, materials handling, painting and even assembly [2–4]. Especially, picking up an object and assembling it to another subsystem accurately is the most important task in an automated manufacturing system [5–7]. To do this task correctly, a robot should be calibrated precisely in advance, and then the robot should be connected with visual sensing systems to observe objects and compute the poses of objects. Although industrial robots generally have high-precision repeatability, the absolute position accuracy of them is not so high due to the kinematic error or assembly tolerance of the robot mechanism [8,9]. In order to improve accuracy, various approaches have been proposed in the literature based on choosing a mathematical error model for robot calibration. From the actual robot experiment, the partial pose or whole pose data of the robot end-effector is gathered, and then it is used to estimate the real robot kinematic parameters. However, the processes of robot calibration considered in the methods are complex, and high precision measurement devices are required to track the nominal pose and the real pose of the robot end-effector. For small-range 3D measurements, touch probes or telescoping-ball bar devices are commonly used. On the other hand, for large-range measurement camera-based systems, coordinate measuring machines (CMM) and laser trackers are typically used [10–14]. However, these types of devices are very expensive to be implemented in robot calibration, and even ineffective.

The conventional robot calibration is often implemented with these convenient devices, but there are still some remaining issues. In fact, the cost of measurement devices and the complexity of the mathematical error model are one of the concerns in the production line. Furthermore, these approaches directly track the pose of the robot end-effector with respect to the device's coordinate and estimate the robot base coordinate by applying the inverse kinematic process. However, this approach does not guarantee high calibration accuracy. In addition, for the vision-based robot system in the industrial field environment, it is essentially desirable that the system is capable of performing calibration without any expensive external apparatus or elaborate setups, which is the system self-calibration [5,6].

Lots of work about robot calibration techniques without using expensive external devices are reported in the literature. Meng et al. [15] proposed a method for robot calibration using vision technology. This approach only requires a ground-truth scale in the reference frame to estimate the pose of the manipulator. However, the proposed method adopts corner detection to extract the corners of the chessboard; its algorithm is easily affected by noise, leading to the failure of corner detection. Then, calibration errors are increased. Gong et al. [16] proposed a method for calibrating and compensating the robot system kinematic error using its internal laser sensor based on distance measurements. However, this approach is restricted by the sensor accuracy not using the absolute position measurement system to measure the robot end-effector. On the other hand, Yin et al. [17] presented an approach for evaluating the kinematic errors of the robot based on the fixed-point constraints to estimate the robot's end-effector. The method is limited to aligning the tool-center-point (TCP) of a robot to fixed points in the robot workspace. The predicted position of the robot's end-effector estimated by the fixed-point and the laser stripe could be misaligned. These techniques are often inconvenient, time consuming and it may not be feasible for some certain applications. To overcome the above limitations, we develop a novel and flexible indirect calibration method for the vision-based robot application that need small position errors. This is a straightforward and efficient method to reduce the absolute position error. Our method does not require the complex solution of the kinematics parameter equations and the complicated procedures of the traditional robot compensation methods. As a result, we successfully reduced the absolute position error the robot's end-effector in the workspace, the position/orientation of the robot's end-effector is compensated without modifying the parameters of the robot. Using the robot's end-effector as an input for a neural network, and the camera attached on the end-effector to observe the object and gather information for training the neural network, the absolute position error of the robot's end-effector is improved. The proposed method is well suited for easy deployment of the robot visual system in the different manufacturing environments because no external measuring equipment or no complicated setup is required in the error compensation and makes the calibration procedure more convenient to implement.

The purpose of our research in this paper is to improve the degree of work in an object picking application by compensating the absolute position error of a six-axis industrial robot by applying a vision-based measurement system. First, the position/orientation of the end-effector was estimated using a vision-based approach combined with the neural network. A novel indirect approach proposed in this paper was used to collect the data in the workspace and to train the neural network, and the calibration methodology among the robot base coordinate, camera system frame and the workspace is described in detail. Then, some simulations and experiments were performed to evaluate the performance of the proposed indirect calibration, and the results were compared in detail to demonstrate the excellence of the proposed method. Finally, some experiments to pick up objects using an industrial robot were conducted to guarantee the positioning performance of the proposed algorithm.

2. Overview of the Problem

The goal of the error compensation approach is to reduce the position error of the robot tool in the real world during the online operation. The added information, usually the real coordinate value of objects, must be precisely determined with respect to the robot coordinate [18]. One can estimate the pose of the object based on its 3D object model known a priori. The pose of the model in the frame (B) is written as

$$ {}^{B}_{W}\boldsymbol{x} = \begin{bmatrix} \boldsymbol{t}_W \\ \boldsymbol{\theta}_W \end{bmatrix}, \tag{1}$$

where $\boldsymbol{t}$ and $\boldsymbol{\theta}$ denote the position and the orientation of the object from the world coordinate to the robot base coordinate, respectively. Based on the ${}^{B}_{W}x$ pose, the transformation matrix describing the model frame (W) relative to the base frame (B) can be determined as follows:

$$ {}^{B}_{W}T = \begin{bmatrix} {}^{B}_{W}R(\theta_W) & {}^{B}_{W}t \\ 0_{1\times 3} & 1 \end{bmatrix}. \tag{2}$$

However, the robot has its own error, so that the real pose of the object corresponding to the robot coordinate is different compared to the ${}^{B}_{W}x$ pose. We denote the real pose as ${}^{B}_{W}x'$. Figure 1 shows that the positions of these two objects differ from the coordinates of the robot.

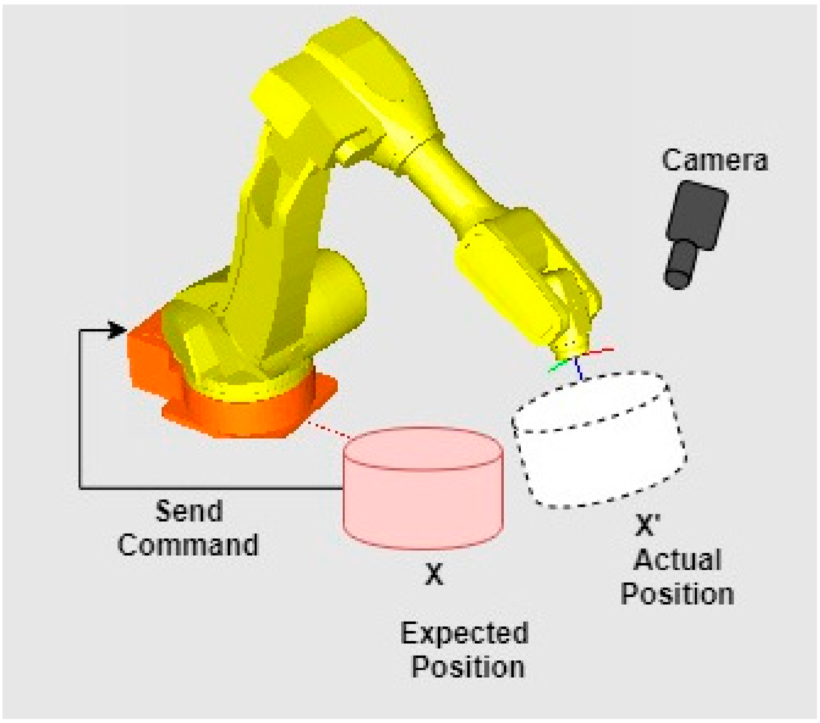


Figure 1. Difference between expected and actual robot position.

In order to pick up objects using the robot, the robot moves to a fine search position. The 3D pose of the correct object is then calculated through the sensor and produces the object coordinates. After calculating the pickup posture considering the 3D posture of the object, the trajectory of the robot is modified to reach the expected position ${}^{B}_{W}x$.

In a real-life scenario, however, the robot will reach an actual location ${}^{B}_{W}x'$ due to a robot's error. From a general point of view, there is a full pose corresponding to the pose ${}^{B}_{W}x$ of an object that allows the controller to control the robot's reach. The important problem is that in order to get to the expected pose correctly, the controller will have to do an error compensation, and the actual pose of the robot will be moved to the expected pose. To solve this problem, we proposed a method using a neural network and a machine vision to predict a new pose ${}^{B}_{W}x_{new}$ after training the data inside the workspace.

Figure 2 describes the architecture of the proposed error compensation algorithm for the object picking system. It consists of two stages: the initial stage and the robot operation stage. The first stage includes the pre-error compensation and error compensation operation. In the pre-error compensation, the operator has to register the reference pattern in the workspace; calibrate the camera in order to detect the 3D pose of the pattern as shown in Block 1. Next, the reference pattern will be detected in the error compensation operation; hand-eye calibration work is implemented to build up the coordinate relationship between camera and robot frames, and the position of the robot will be estimated in this operation based on the neural network as shown in Block 2. Finally, the whole process of object picking is described in the robot operation stage, which includes the go-to-fine-search-position to pick-object as shown in Block 3. The main technical functions of the proposed robot error compensation are training the workspace for the robot using the indirect calibration approach.

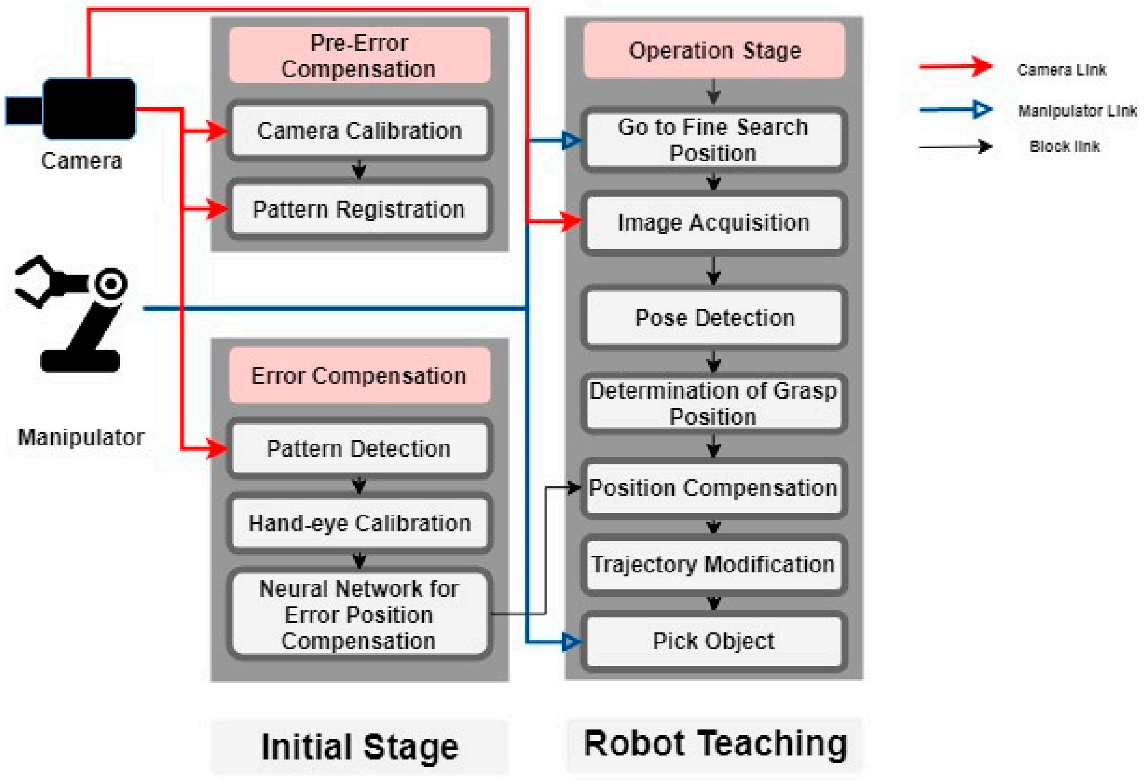


Figure 2. The overall architecture of proposed error compensation algorithm.

3. Proposed Error Compensation Method

3.1. Pattern Detection and Pose Estimation

In this study, the world coordinate system is located on the pattern board, which is the calibrated workspace for the robot. For simple detection and pose estimation of the pattern board, we designed a specific pattern image as seen in Figure 3a. The circles on the pattern board are numbered as seen in Figure 3d, four bigger circles. The pattern board consists of the key feature made of four big circles, which are used to create the workspace coordinate, and the rest of the circles with smaller size are used for the pose estimation. The operation of the pre-compensation system is divided into two stages: the pattern detection and pose estimation. For pattern image detection we moved the robot to the pattern board, then the hole geometry was extracted from the image. In the pose estimation stage, the Perspective-n-Point (PnP) method based on the RANSAC-algorithm was used to estimate the coordinate of the pattern corresponding to the camera system.

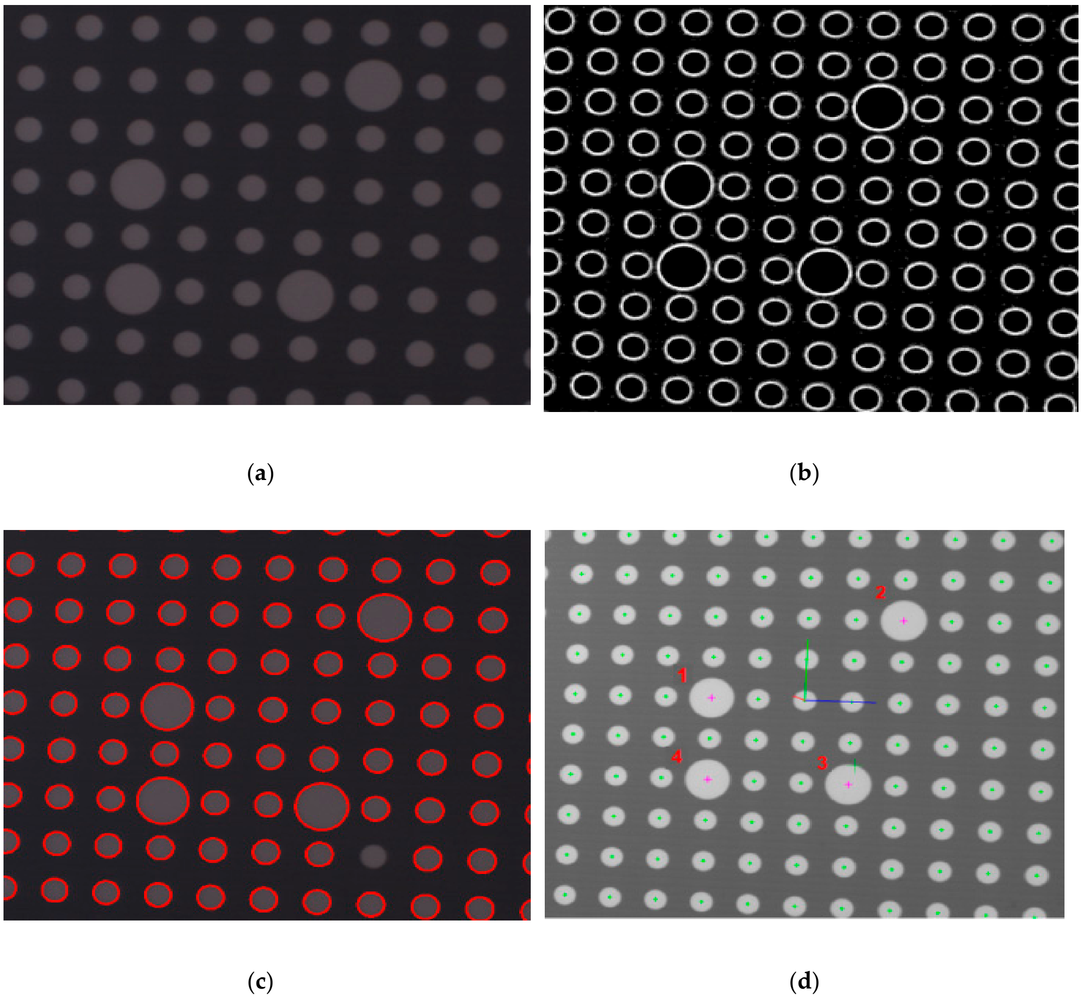


Figure 3. Pattern detection and pose estimation. (**a**) Original image, (**b**) edge detection, (**c**) ellipse fitting, (**d**) pose detection.

In the pattern detection stage, we extracted the holes section from image data via the edge detection algorithm as shown in Figure 3b. We applied a Difference-of-Gaussian (DoG) filter to remove noise and improve the geometry shape of the holes. DoG is accomplished by convolving the image with a Gaussian filter at different scales. As a result, key-points were extracted from multiple scales, based on the maximum/minimum value of DoG:

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)\ , \tag{3}$$

where $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$.

In order to increase the reliability of pattern detection, the ellipse fitting techniques were additionally applied for circle detection as shown in Figure 3c. The ellipse-fitting algorithm is a common task in machine vision to estimate the centers and radius of the circle. Let denote a set of 2D data points $P = \{x_i\}_{i=1}^{n},\ where\ \boldsymbol{x}_i = (x_i, y_i)$, a family of curves C(**a**) parameterized by the vector **a**, and the distance metric $\delta(C(\boldsymbol{a}), \boldsymbol{x})$ which measures the distance from a point **x** to the curve $C(\boldsymbol{a})$. The problem is to find the value $\boldsymbol{a}_{min}$ for which the error function $\epsilon^2(\boldsymbol{a}) = \sum_{i=1}^{n} \delta(C(\boldsymbol{a}), \boldsymbol{x_i})$ attains its global minimum. Hence, the curve best fits the data. In this study, the fitting algorithm based on the "approximate mean square distance" metric [19], minimizes the unusual objective function,

$$\epsilon^2(a) = \frac{\sum_{i=1}^{n} F(a, \boldsymbol{x}_i)^2}{\sum_{i=1}^{n} \parallel \nabla_x F(a, \boldsymbol{x}_i) \parallel^2} = \frac{\parallel D\boldsymbol{a} \parallel^2}{\parallel D_x\boldsymbol{a} \parallel^2 + \parallel D_y\boldsymbol{a} \parallel^2}\ , \tag{4}$$

where the matrices D_x and D_y are the partial derivatives of D with respect to x and y.

Based on the detection stage, key-points localization is suitably implemented using a pose estimation algorithm. To estimate the pose of the pattern, many approaches have been proposed in the literature. For this work, we applied the RANSAC algorithm to estimate the pose. RANSAC is an iterative method proposed to solve the PnP problem [20]. Since then, it has been applied to many machine vision areas such as PnP, visual SLAM, homographic estimation, fundamental or essential matrix estimation, etc. Let assume we have a set of pairs of matched 2D-3D points (corresponding):$\left(x_i,\ X_i^W\right)$, as shown in Figure 3d, four major feature points are used to define the workspace coordinate. The final solution is the transformation matrix T_W^C that transforms from the workspace coordinate to camera coordinate.

Since we applied robust image processing algorithms such as pattern detection and ellipse fitting to increase the performance of 3D position and orientation, the proposed system becomes more reliable on data preparation for the neural network used to compensate the error position.

3.2. The Hand-Eye Calibration

The initial vision-based robot involved determining the coordinate relationship between the robot coordinate and the sensor coordinate [21,22]. Common setup of the sensor can be located at a fixed position or can be mounted on the tool of the robot according to a specific application. In this work, the sensor was attached to the tool of the robot for a better field of view in the workspace. The setup of the sensor was fixed on that position during the calibration. If there was any rearrangement about the setting, the calibration was implemented again.

Figure 4 represents the coordinate frames used to perform the hand-eye calibration in this paper, where (B), (E), (C) and (W) are the coordinates of the robot base, the robot, the camera, and the world, respectively. The relationship between each coordinate can be described by a homogenous transformation matrix.

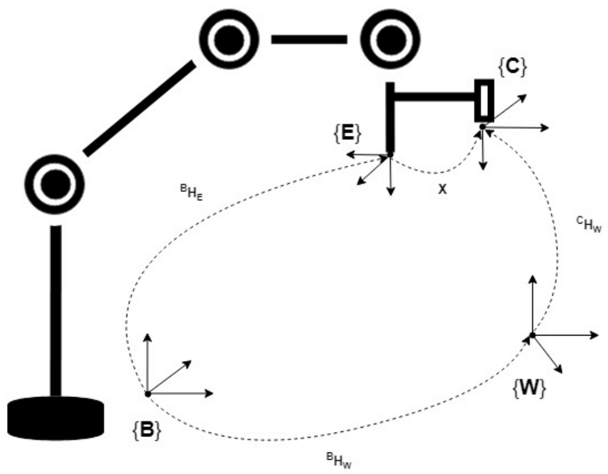


Figure 4. Coordinates for hand-eye calibration.

In the literature, several approaches have been published to solve the hand-eye calibration; the problem yields a homogeneous matrix equation of the $\boldsymbol{AX} = \boldsymbol{XB}$ form as the following:

$$\boldsymbol{r}_a\boldsymbol{r}_x = \boldsymbol{r}_x\boldsymbol{r}_b\ , \tag{5}$$

$$(\boldsymbol{r}_a - \boldsymbol{i}_3)\boldsymbol{t}_x = \boldsymbol{r}_x\boldsymbol{t}_b - \boldsymbol{t}_a\ , \tag{6}$$

where $\boldsymbol{i}_3$ is 3×3 unit matrix $\boldsymbol{r}_a, \boldsymbol{r}_b \in$ SO3 are rotation matrices corresponding to the robot and camera transformation, respectively. $\boldsymbol{r}_x\ \in SO3\ , t_x\ \in R^3$ are the rotation matrix and translation matrix, respectively, which denotes $\boldsymbol{X}$. The linear optimization method is a common solution to solve this equation based on the assumption that $\boldsymbol{A}$, $\boldsymbol{B}$ satisfy the rigid transformation or their rotation angles in Equation (5) are equal. However, in most cases, their $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{X}$ might not satisfy rigid transformation. A direct solution based on the iterative computation with Jacobian optimization is proposed by Jianfei et al. [23]. Given multiple pairs, $(\boldsymbol{A}_i,\ \boldsymbol{B}_i)\ for\ i = 1, \ldots, n$, where i represents the sequence number of the equation $\boldsymbol{AX} = \boldsymbol{XB}$. The problem of hand-eye can be stated using the properties of Kronecker product ($\otimes$), the Equations (5) and (6) can be written as:

$$F(i) = \left(\boldsymbol{r}_a(i) \otimes \boldsymbol{i}_3 - i_3 \otimes \boldsymbol{r}_b^t(i)\right)\boldsymbol{vec}(\boldsymbol{r}_X)\ , \tag{7}$$

$$G(i) = (\boldsymbol{r}_a(i) - \boldsymbol{i}_3)\boldsymbol{t}_{ai} - \boldsymbol{t}_X\boldsymbol{t}_{bi} + \ \boldsymbol{t}_{ai}\ , \tag{8}$$

where vec is an operation, which stretches a matrix as the row's direction, $vec(R_X) \in \mathbb{R}^9$ is the vector obtained by stacking the columns of $\boldsymbol{t}_X$, and $F(i)$, $G(i)$ are the vectors of (9×1) and (3×1) separately. Find the rotation $\boldsymbol{\theta}_{min} \in R^3$ and the translation $\boldsymbol{t}_{min} \in R^3$ for which the error function $L(\boldsymbol{\theta},\ \boldsymbol{t}) = \min \sum_i [\|\ F(i)\ \|^2 + \|\ G(i)\ \|^2]$ attains its global minimum, where $L(\boldsymbol{\theta},\ \boldsymbol{t})$ is the objective function of optimization for solving the hand-eye problem. Let $J(i)$ be the Jacobian formula for the object function, $H = \ [F(i), G(i), \ldots, F(n), G(n)]^T$ can be represented of multi-equations such as Equations (7) and (8). Then the interactive formula for optimization is given as:

$$\boldsymbol{J}\Delta_X = \ -\boldsymbol{H}\ , \tag{9}$$

$$\boldsymbol{X}^{\boldsymbol{n}+\boldsymbol{1}} = \ \boldsymbol{X}^{\boldsymbol{n}} + \Delta_X\ . \tag{10}$$

The transformation $\boldsymbol{X}(\boldsymbol{\theta}, \boldsymbol{t})$ is then the solution that *best fits* the multiple pairs $(\boldsymbol{A}_i,\ \boldsymbol{B}_i)\ for\, i = 1, \ldots, n$. A linear solution can be found using singular value decomposition (SVD) or using a pseudo-inverse.

After solving the hand-eye problem, the object position in the world coordinate corresponding to the robot coordinate can be described as follows:

$$\boldsymbol{H}_W^B = \ \boldsymbol{H}_E^B\boldsymbol{X}\boldsymbol{H}_W^C\ , \tag{11}$$

$$\boldsymbol{H}_i^B = \ \boldsymbol{H}_W^B\boldsymbol{P}_i^W\ , \tag{12}$$

where $\boldsymbol{X}$ is obtained from the work above, $\boldsymbol{H}_E^B$ is provided by the robot controller, $\boldsymbol{H}_W^C$ is calculated by the pattern detected from the camera, and $\boldsymbol{P}_i^W$ is the 3D position of the object in world coordinate.

3.3. Feature Training Using Neural Network

A non-parametric kinematics calibration is an approach using intelligent algorithms to reduce the position error without modifying robot parameters [9]. Its advantage is that the position is compensated directly, by which the calibration process can be simplified. Several approaches to error compensation without robot parameter modification are introduced in the literature [24,25]. The real position of the robot can be tracked by using external devices such as a laser tracker, stereo vision, etc. Then, the non-linear approach is used to estimate the error between the actual position and the real position to minimize the error of the robot.

In this paper, we proposed an error compensation method based on advanced machine-vision algorithms and neural network to guide the robot to pick the object. Figure 5 shows the difference between the direct and the indirect approach presented in this paper. When applying the indirect approach, the robot's end-effector is set to be interpolated, assuming that the coordinate conversion value from the measuring object to the camera coordinate system is obtained accurately. The transformation from the camera coordinate to the root always has a constant value. In addition, the real position of the end-effector and commanded position ${}_E^B\boldsymbol{P}_a$ from the robot controller is different, so the actual position of the robot controller in the world coordinate frame is defined as:

$${}_E^B\boldsymbol{P}_a = \boldsymbol{H}_W^B\boldsymbol{H}_C^W\boldsymbol{X}^{-1}\ , \tag{13}$$

where $\boldsymbol{X}$, $\boldsymbol{H}_W^B \in SE(3)$ are obtained from the above, and $\boldsymbol{H}_C^W \in SE(3)$ is calculated from the camera. To collect information from the workspace, the robot moves to all defined 3D grids to obtain images from the camera to determine the position of the object, where the 3D grid is the workspace where the robot will be trained during the error-compensation stage; finally, the actual position of the robot is calculated by the controller and stored in memory. This data is used for training purposes. Figure 6 shows our proposed method.

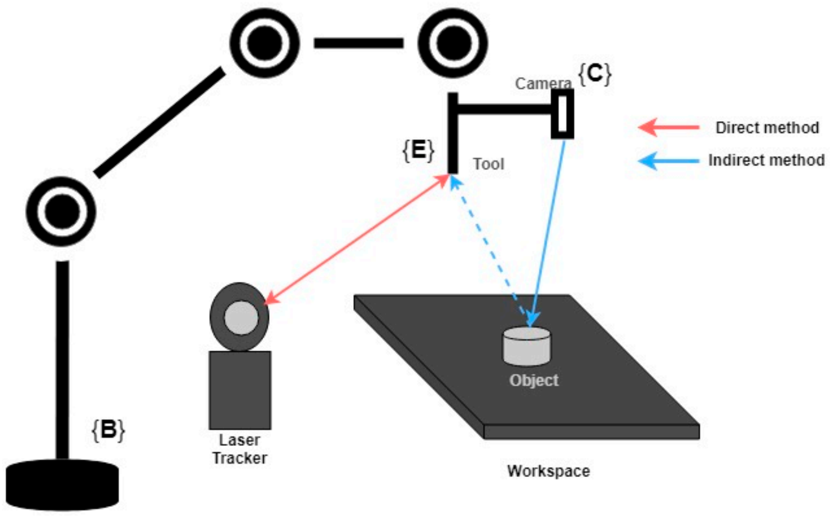


Figure 5. An indirect method for robot calibration.

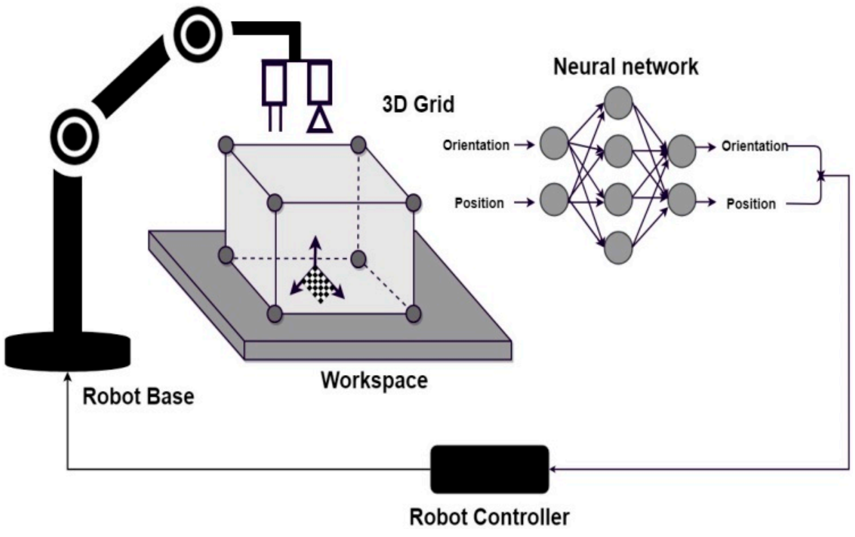


Figure 6. Indirect method for robot error-compensation.

The next step is to perform training using measured data. In this paper, we applied a neural network to estimate the actual location of the robot. Neural networks are currently the state of the art method in machine learning [26].

In our experiment, we used a three-layer feedforward neural network to classify different types of defects. Let $V = \left(V_1^T, V_2^T, \ldots, V_m^T\right)^T$ be the weight matrix connecting the input and the hidden layers, where $V_j = \left(V_{j1}, \ldots, V_{jn}\right)$ for $j = 1, 2, \ldots, m$. Let $W = \left(W_1^T, W_2^T, \ldots, W_p^T\right)^T$ be the weight matrix between the hidden and the output layers, where $W_k = (W_{k1}, W_{k2}, \ldots, W_{km})$ for $k = 1, 2, \ldots, p$. Given input $x = (x_1, \ldots, x_n)^T \in R^n$, and the final output vector $o = \left(o_1, \ldots, o_p\right)^T \in R^p$ is given:

$$o_k = f(W_k.y - b_k) = f\left(\sum_{j=1}^{m} W_{kj}y_j - b_k\right), \quad k = 1, \ldots, p. \tag{14}$$

where $\{b_k\}_{k=1}^{p}$ is the biases from hidden to output layers, and $\boldsymbol{y} = (y_1, \ldots, y_n)^T \in R^m$ is the output of the hidden layer. The distance error is measured based on mean square error, defined as

$$E(W, V) = \frac{1}{2}\sum_{h=1}^{H} \|z_h - o_h\|^2, \tag{15}$$

where $\boldsymbol{z_n} = (z_1, \ldots, z_n)^T \in R^n$ is the desired output from the dataset. The update rules based on gradient descent for the weight vector are

$$\mathrm{W}_{\mathrm{kj}}^{(\mathrm{l}+1)} = \mathrm{W}_{\mathrm{kj}}^{(\mathrm{l})} + \eta\frac{\delta\mathrm{E}\left(\mathrm{W}^{(\mathrm{l})}, \mathrm{V}^{(\mathrm{l})}\right)}{\delta\mathrm{W}_{\mathrm{kj}}}, \tag{16}$$

$$\mathrm{V}_{\mathrm{kj}}^{(\mathrm{l}+1)} = \mathrm{V}_{\mathrm{kj}}^{(\mathrm{l})} + \eta\frac{\delta\mathrm{E}\left(\mathrm{W}^{(\mathrm{l})}, \mathrm{V}^{(\mathrm{l})}\right)}{\delta\mathrm{V}_{\mathrm{kj}}}. \tag{17}$$

where $l = 0, 1, 2, \ldots$; $k = 1, 2, \ldots, p$; $j = 1, 2, \ldots, m$; and $i = 1, 2, \ldots, n$.

4. Simulation for Robot Model

To improve errors based on the direct approach, the simulation was performed based on PUMA robot parameters before and after compensation using a neural network. For the comparison, two robot models were created: the nominal model and the error model. All DH parameters of the error model added to the nominal model with uniformly distributed noise. Let ε_{length} and $\varepsilon_{angular}$ be the length error magnitude (mm) and angular error offset (deg). The noises added to the nominal model parameters with length error $\Delta\varepsilon_{length}$(mm) and angular error $\Delta\varepsilon_{angular}$(deg) are given as:

$$\Delta\varepsilon_{length} = G(x_l, \sigma_l) * \varepsilon_{length}, \tag{18}$$

$$\Delta\varepsilon_{angular} = G(x_a, \sigma_a) * \varepsilon_{angular}, \tag{19}$$

where $G(\mathrm{x}\bullet, \sigma\bullet) = \frac{1}{2\pi\sigma\bullet^2}e^{-x\bullet^2/2\sigma\bullet^2}$, $\mathrm{x}\bullet, \sigma\bullet$ are the mean and standard deviation. The geometric errors in the robot-link could be written as $\Delta a_i, \Delta\alpha_i, \Delta d_i, \Delta\theta_i$ while the original parameters of the robot-link i^{th} were denoted as $a_i, \alpha_i, d_i, \theta_i$. The parameters of error model are as following: $a_i{}^r = a_i + \Delta a_i\ (mm)$, $\alpha_i{}^r = \alpha_i + \Delta\alpha_i\ (mm)$, $d_i{}^r = d_i + \Delta d_i$ (mm).

Where $\Delta\bullet$ is the noise determined in Equations (18) and (19). The actual values for two kinematic models are shown in Tables 1 and 2, respectively. In this simulation, $\varepsilon_{length} = 10$ mm, and $\varepsilon_{angular} = 2deg$.

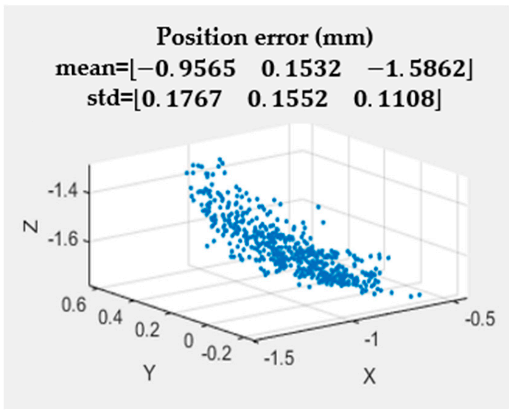
Table 1. DH parameters of the nominal model.

Link No.	θ_i/rad	d_i/mm	a_i/mm	α_i/rad
1	θ_1	0	0	1.5708
2	θ_2	0	432	0
3	θ_3	150	20	−1.5708
4	θ_4	432	0	1.5808
5	θ_5	0	0	−1.5708
6	θ_6	0	0	0

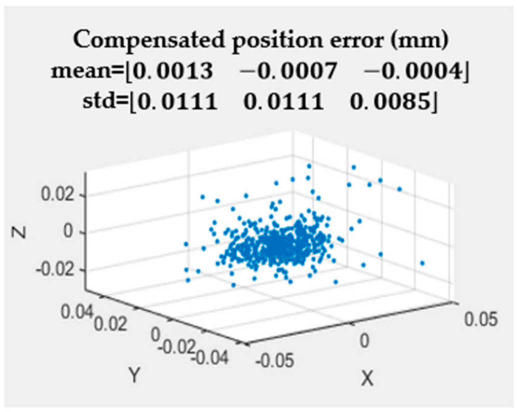
Table 2. DH parameters of the error model.

Link No.	θ_i/rad	d_i/mm	a_i/mm	α_i/rad
1	θ_1	2.42463	3.433	1.65256
2	θ_2	5.92042	433.966	0.031139
3	θ_3	150.023	20.0033	−1.441
4	θ_4	436.297	0.317539	1.5975
5	θ_5	5.57796	1.30921	−1.4774
6	θ_6	4.17643	6.27515	0.1111

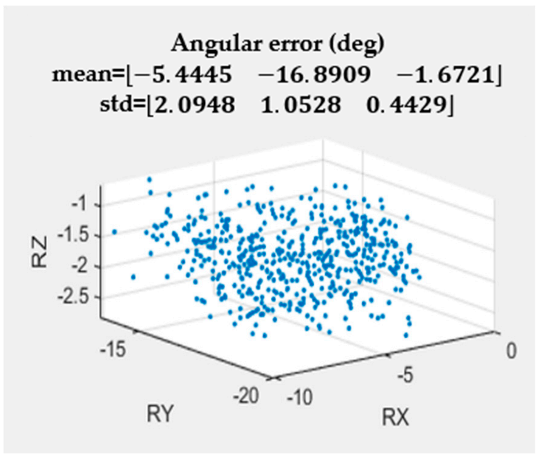
Firstly, the simulation was based on the neural network combined with the laser tracker system. The simulation describes the use of an artificial neural to estimate the robot's end-effector where the actual positions of end-effector in the workspace are known. The robot is moved to all 3D grid points, all position errors in the 3D grid are measured and recorded by the laser tracker system, and these position errors are stored in the memory to train the neural network. The simulation results are shown in Figure 7. The errors have been improved after the compensation. However, a direct approach using a laser tracker is not well suited for vision-based robot applications because of the cost-effectiveness of the external measuring device and the elaborate setup for the calibration procedure.



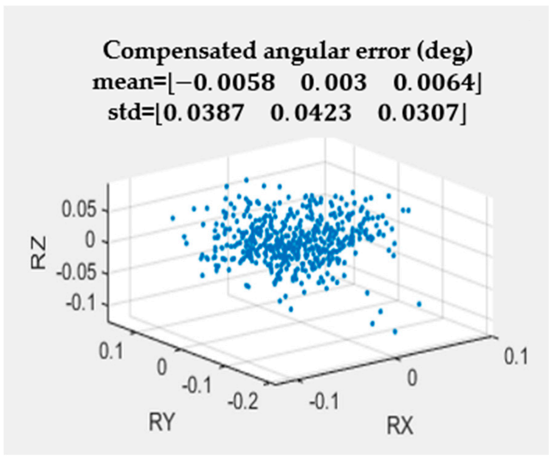
(**a**)



(**b**)



(**c**)



(**d**)

Figure 7. The error compensation results. (**a**) position error (mm) (**b**) compensated position error (mm) (**c**) angular error (deg) (**d**) compensated angular error (deg).

Secondly, to illustrate the validity of the proposed method, this section performs error compensation in the simulation environment of the PUMA 560 robot model. For comparison, two robot models were created the same as above. The data generated in this simulation for solving the hand-eye vision problem and training the neural network is also described in this section.

4.1. Simulation Procedure

The procedure for the simulation process is divided into two phases. In the first phase, data is collected from both models. An error compensation assessment is then performed. For the first step, for the hand-eye calibration ($\boldsymbol{AX} = \boldsymbol{XB}$), real data for the transformation matrix $\boldsymbol{A}_i$ was generated by the nominal robot model and data for the transformation matrix $\boldsymbol{B}_i$ was generated by the robot error model. Next, in the training for the neural network section, m *(m = 686)* set of samples were used, and 200 random samples on various positions and orientations in the work coordinate frame were examined for testing. In addition, the distance between the neighboring of each grid points was 28.5 (mm) in all three directions X, Y, and Z, which is an empirical interval for a mid-size calibration space. In total, the workspace includes 343 cells ($7 \times 7 \times 7$ mm^3). At each cell, two different orientations were taken.

In the error compensation stage, a generalized feed-forward neural network is used. This network consists of one hidden layer. As presented in Figure 8, there are 50 neurons in the hidden layer. The desired position and orientation $\boldsymbol{x} = (x_1, \ldots, x_6)^T \in R^6$ of the robot's end-effector is taken as input layer nodes, and the related position/orientation $\boldsymbol{o} = (o_1, \ldots, o_6)^T \in R^6$ for robot controller is taken as the output layer nodes.

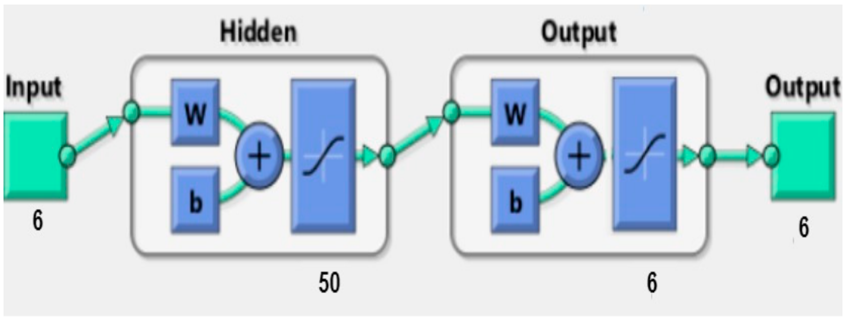


Figure 8. Architecture developed for the neural network.

4.2. Simulation Results

After training the neural network, we used 200 test data to evaluate the performance of position/orientation error, and the results are shown in Figure 9.

In Figure 9, the blue area is the result of the robot system before training, while the red area is the result after applying neural network training. The position error $\boldsymbol{t_{Error}} = \left(t_x,\ t_y, t_z\right)^T \in R^3$ and $\boldsymbol{\theta_{Error}} = \left(\theta_x,\ \theta_y, \theta_z\right)^T \in R^3$ are defined as follows:

$$\boldsymbol{t_{Error}} = \boldsymbol{t_{estimated}} - \boldsymbol{t_{real}}\,, \tag{20}$$

$$\boldsymbol{\theta_{Error}} = \boldsymbol{\theta_{estimated}} - \boldsymbol{\theta_{real}}\,. \tag{21}$$

As you can see in Figure 9, the errors have been greatly reduced when using the method proposed in this paper. In addition, you can see that applying the neural network to the indirect compensation approach significantly reduces the mean and standard deviation of error.

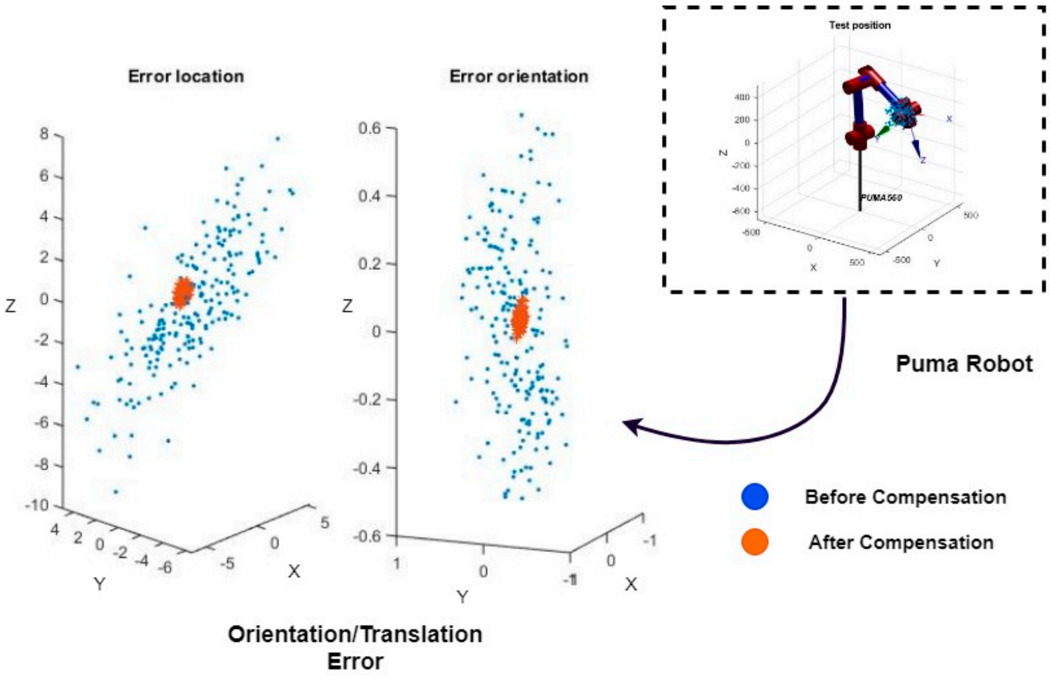


Figure 9. Position/orientation error in 3D space for simulation.

Table 3 shows that the mean error decreased significantly when the error was compensated using the neural network compared to when the error was not compensated. The mean position errors after compensation are $e_{t_x} = -0.0295$ (mm), $e_{t_y} = -0.0079$ (mm), and $e_{t_z} = -0.0496$ (mm), for which the errors are reduced by 98% on average. As shown in Table 4, the standard deviation for position/orientation error was greatly reduced: $e_{t_x} = 0.3583$ (mm), $e_{t_y} = 0.5101$ (mm), and $e_{t_z} = 0.03634$ (mm), approximately 94% of error reduction on average.

Table 3. Mean error of the measurement system position-accuracy.

Measurement	**Mean Error**					
	Translation/mm			**Rotation/rad**		
	e_{t_x}	e_{t_y}	e_{t_z}	e_{θ_x}	e_{θ_y}	e_{θ_z}
Before	0.9883	−1.3173	2.3743	0.0008	−0.0062	−0.0032
After	−0.0295	−0.0079	−0.0496	−0.0000	0.0002	0.0000
Reduced %	97.01%	99.4%	97.9%	100%	96.77%	100%

Table 4. Standard deviation error.

Measurement	**Standard Deviation Error**					
	Translation/mm			**Rotation/rad**		
	e_{t_x}	e_{t_y}	e_{t_z}	e_{θ_x}	e_{θ_y}	e_{θ_z}
Before	12.4182	10.4763	8.7984	0.0202	0.0215	0.0171
After	0.2632	0.4256	0.3881	0.0006	0.0006	0.0006
Reduced %	97.88%	95.93%	95.58%	97.02%	97.21%	96.49%

5. Experimental Results

We used a Hyundai Hi5 (HA006 model) 6-axis industrial robot to conduct the experiment. A high-resolution (12 Mp) baser camera with a focal length of 8 mm was attached to the end effector of the robot. In addition, a pneumatic gripper was attached to the robot's end-effector to enable the robot to grasp objects. The overall robot system is shown in Figure 10.

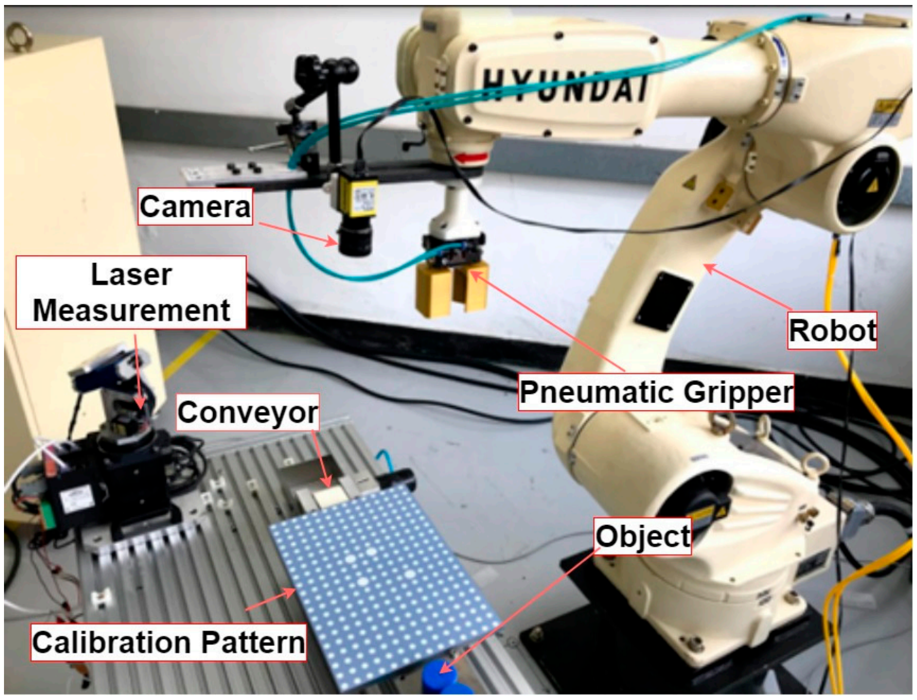


Figure 10. Setup for training robot.

5.1. *Experiments on Position/Orientation Error*

To evaluate the performance of the proposed system, we compared the calculated amount of movement of the robot to the actual amount of movement. The amount of robot movement can be easily calculated using Equation (13). In this error evaluation process, 686 data samples were used for neural network training, and 200 data samples were used for testing the neural network. The test results are shown in Figure 11. Although the actual results are not as good as the results from the simulation environment, they have reduced the position/orientation error sufficiently. The after compensation value of mean error-position in each direction as shown in Table 5 are $e_{t_x} = -1.3897$ (mm), $e_{t_y} = -2.4289$ (mm), and $e_{t_z} = 1.554$ (mm), for which the error was reduced by 50.3% on average.

In Table 6, the standard deviation for position/orientation errors after compensation are greatly reduced: $e_{t_x} = 0.6998$ (mm), $e_{t_y} = 0.8826$ (mm), and $e_{t_z} = 0.4484$ (mm), approximately 69% of error reduction on average. The proposed method showed good performance as a result of the experiment. In Figure 11, the after-compensation error (red line) is smaller and smoother than the before compensation error (blue line) when applying the error compensation technique. This illustrates that the absolute position error of the robot's end-effector is improved. Considering the calculated data from the simulation and experiment as shown in Tables 3–6, respectively, the compensation improvement of the experimental cases is smaller than the result of the simulation cases. The main reason is that it is really difficult in the real experiment to consider all factors that lead to the absolute position error of the robot's end-effector such as tolerances, eccentricities, wear-out, payload, temperature and insufficient knowledge of model parameters for the transformation between robot poses, etc. [5]. However, our algorithm did reduce the absolute position error in real experiments by 50.3%, which can verify the proposed compensation algorithm to be utilized successfully in the real application.

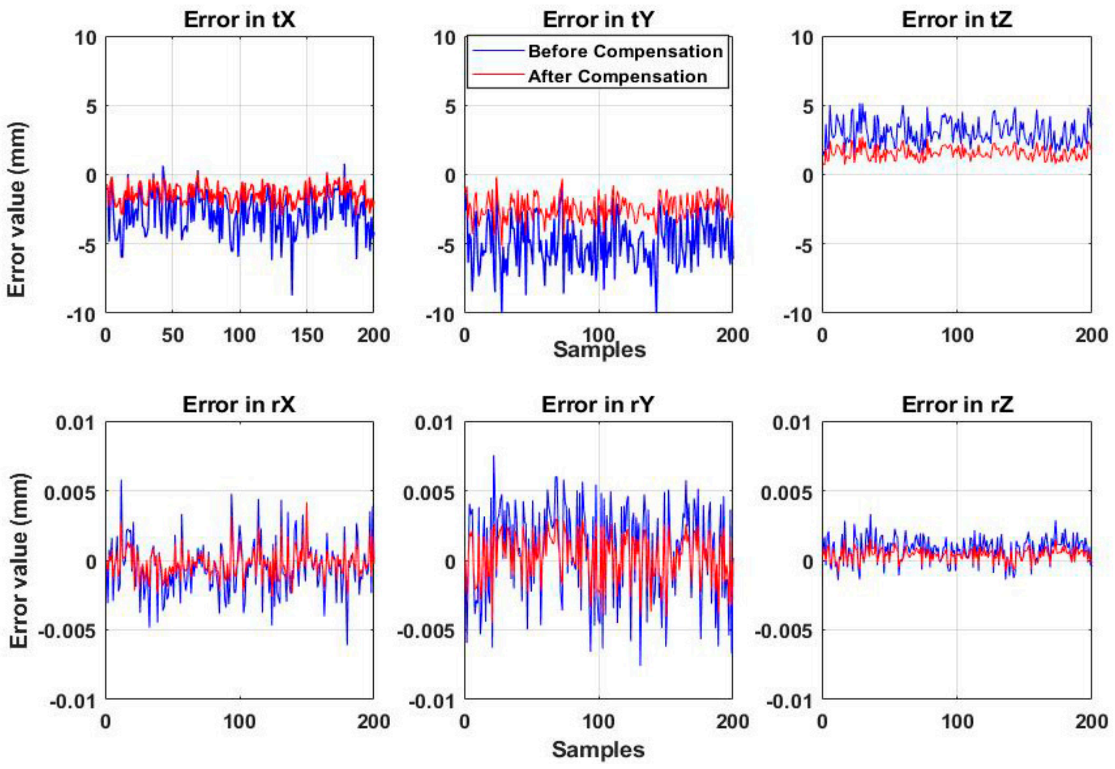


Figure 11. Comparison between before and after compensation in the experiment environment.

Table 5. Mean error of the measurement system position accuracy.

Measurement	**Mean Error**					
	Translation/mm			**Rotation/rad**		
	e_{t_x}	e_{t_y}	e_{t_z}	e_{θ_x}	e_{θ_y}	e_{θ_z}
Before	−2.9269	−4.9840	2.9249	−0.0004	0.0007	0.0008
After	−1.3897	−2.4289	1.5540	−0.0002	0.0004	0.00045
Reduced %	52.52%	51.27%	46.87%	50%	42.86%	43.75%

Table 6. Standard deviation error.

Measurement	**Standard Deviation Error**					
	Translation/mm			**Rotation/rad**		
	e_{t_x}	e_{t_y}	e_{t_z}	e_{θ_x}	e_{θ_y}	e_{θ_z}
Before	2.2461	2.3726	1.7413	0.0019	0.0036	0.0010
After	0.6998	0.8826	0.4484	0.0010	0.0018	0.00055
Reduced %	68.84%	62.80%	74.25%	47.37%	50%	45%

Comparing the compensation performance the other works, Liu et al. [27] proposed an improved method for the pose accuracy of the robot manipulator by using a multiple-sensor combination measuring system (MCMS). In their experiments, the pose accuracy of the manipulator is improved by 67.3%, to 3.379 mm on average with the Kamal filter (KF) and by 38.2%, to 1.286 mm on average with multi-sensor optimal information fusion algorithm (MOIFA). Yauheni et al. [2] proposed a method of robot end-effector pose accuracy improving using joint error mutual compensation, the improved value of positioning accuracy for the robot end-effector from 2% to two times, to ΔL = 2.39 mm on average. Hence, it can be concluded that our proposed method gives a better performance on the whole, both in terms of the error reduction ratio and the absolute position error, which is a quite acceptable error in robot application in the real field.

5.2. The Qualitative Experiments Results

To verify the validity of the error compensation method proposed in this paper, an object-picking task was performed using the robot used in the experiment. The procedure for the object picking task is described in Figure 2. Using C#, control software was developed and Raspberry board was used for communication with the robot. Communication was carried out using the RS-232 standard. First, the robot was moved to the fine search position in the field-of-view area of the camera. Then, the camera was used to calculate the 3D pose of an object and then combined with the information of the hand-eye calibration. Then, the position and the normal vector of the object was calculated. Next, the position/orientation value of the end effector was estimated using a neural network. Finally, the robot's trajectory was modified, and the robot's gripper reached the target object. The photo of the robot's picking task is shown in Figures 12 and 13.

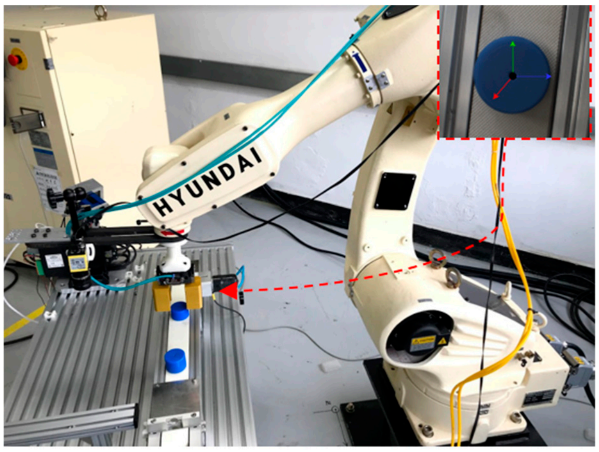


Figure 12. Trajectory that the robot was taught.

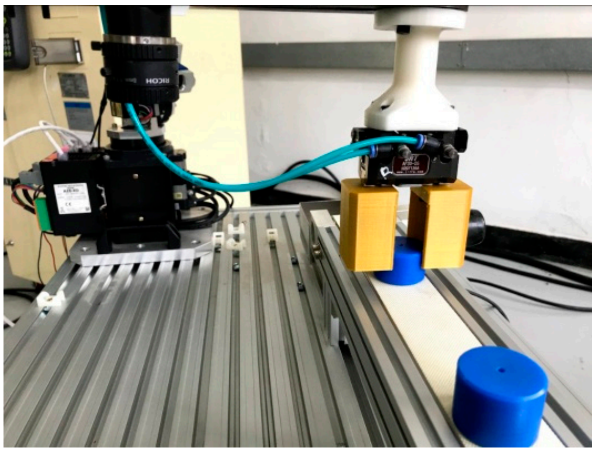


Figure 13. Object picking result.

The results of the experiment confirmed that the proposed method reduced the absolute position error in the workspace with 200 random positions by 50.3% on average, which is sufficiently applicable to the object-picking task. We are confident that applying the method proposed in this paper to robots will also allow for tasks requiring higher accuracy.

6. Conclusion

In this paper, the proposed indirect calibration approach is proved to compensate for the absolute position/orientation error of a six-axis industrial robot throughout the simulations and experiments. In particular, experiments with an object picking task using a robot and camera were also conducted to substantively demonstrate the validity of the proposed algorithm. The position/orientation of the robot's end-effector is compensated without modifying the robot's parameters. The proposed method is based on a machine vision algorithm combined with a neural network. Using the robot's end-effector as an input for a neural network, and the camera attached on the end-effector to observe the object, we successfully improved the absolute position error of the robot in the workspace. According to the simulation results, location errors decreased by 98%. The average value of the absolute position error was 0.029 mm. Actual results showed that the absolute position error was reduced to 50.3% and the average value of the absolute position error was 1.79 mm, which is a quite acceptable error in robot application in the real field [5–7]. In conclusion, the proposed method is well suited for simply deploying the robot visual system in different manufacturing environments because no external measuring equipment or complicated setup is required in the error compensation and makes the calibration procedure more convenient to implement. We believe that the method proposed in this paper can also be applied to robot tasks requiring a high degree of accuracy and replace the existing error-compensating methods.

Author Contributions: All authors read and approved the manuscript. Conceptualization, C.-T.C.; methodology, C.-T.C. and V.-P.D.; Software, C.-T.C. and V.-P.D.; validation, C.-T.C. and B.-R.L.; format analysis, C.-T.C. and B.-R.L.; investigation, C.-T.C.; resource, C.-T.C.; Writing—Original Draft preparation, C.-T.C.; Writing—Review and Editing, B.-R.L.; visualization, C.-T.C.; supervision, B.-R.L.

Acknowledgments: This work was supported by the 2019 Research Fund of University of Ulsan.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Oh, J.-K.; Lee, S.; Lee, C.-H. Stereo vision based automation for a bin-picking solution. *Int. J. Control Autom. Syst.* **2012**, *10*, 362–373. [CrossRef]
2. Yauheni, V.; Jerzy, K. Application of joint error mutual compensation for robot end-effector pose accuracy improvement. *J. Intell. Robot. Syst. Theory Appl.* **2003**, *36*, 315–329.
3. Darmanin, R.N.; Bugeja, M.K. A review on multi-robot systems categorised by application domain. In Proceedings of the 2017 25th Mediterranean Conference on Control and Automation (MED), Valletta, Malta, 4–6 July 2017; pp. 701–706.
4. Njaastad, E.B.; Egeland, O. Automatic Touch-Up of Welding Paths Using 3D Vision. *IFAC-PapersOnLine* **2016**, *49*, 73–78. [CrossRef]
5. Pérez, L.; Rodríguez, Í.; Rodríguez, N.; Usamentiaga, R.; García, D. Robot Guidance Using Machine Vision Techniques in Industrial Environments: A Comparative Review. *Sensors* **2016**, *16*, 335. [CrossRef] [PubMed]
6. Labudzki, R.; Legutko, S. Applications of Machine Vision. *Manuf. Ind. Eng.* **2011**, *2*, 27–29.
7. Wöhler, C. *3D Computer Vision: Efficient Methods and Applications*; Springer: Dortmund, Germany, 2009.
8. Roth, Z.; Mooring, B.; Ravani, B. An overview of robot calibration. *IEEE J. Robot. Autom.* **1987**, *3*, 377–385. [CrossRef]
9. Xuan, J.Q.; Xu, S.H. Review on kinematics calibration technology of serial robots. *Int. J. Precis. Eng. Manuf.* **2014**, *15*, 1759–1774. [CrossRef]
10. Nubiola, A.; Bonev, I.A. Absolute calibration of an ABB IRB 1600 robot using a laser tracker. *Robot. Comput. Integr. Manuf.* **2013**, *29*, 236–245. [CrossRef]
11. Nubiola, A.; Bonev, I.A. Absolute robot calibration with a single telescoping ballbar. *Precis. Eng.* **2014**, *38*, 472–480. [CrossRef]
12. Kubota, T.; Aiyama, Y. Calibration of relative position between manipulator and work by Point-to-face touching method. In Proceedings of the 2009 IEEE International Symposium on Assembly and Manufacturing, Suwon, Korea, 17–20 November 2009; pp. 286–291.

13. Bai, Y.; Zhuang, H.; Roth, Z.S. Experiment study of PUMA robot calibration using a laser tracking system. In Proceedings of the 2003 IEEE International Workshop on Soft Computing in Industrial Applications, Binghamton, NY, USA, 25 June 2003; pp. 139–144.
14. Ha, I.-C. Kinematic parameter calibration method for industrial robot manipulator using the relative position. *J. Mech. Sci. Technol.* **2008**, *22*, 1084–1090. [CrossRef]
15. Meng, Y.; Zhuang, H. Autonomous robot calibration using vision technology. *Robot. Comput. Integr. Manuf.* **2007**, *23*, 436–446. [CrossRef]
16. Gong, C.; Yuan, J.; Ni, J. A Self-Calibration Method for Robotic Measurement System. *J. Manuf. Sci. Eng.* **2000**, *122*, 174–181. [CrossRef]
17. Yin, S.; Ren, Y.; Zhu, J.; Yang, S.; Ye, S. A Vision-Based Self-Calibration Method for Robotic Visual Inspection Systems. *Sensors* **2013**, *13*, 16565–16582. [CrossRef] [PubMed]
18. Chang, W.-C.; Wu, C.-H. Eye-in-hand vision-based robotic bin-picking with active laser projection. *Int. J. Adv. Manuf. Technol.* **2016**, *85*, 2873–2885. [CrossRef]
19. Fitzgibbon, A.; Fisher, R. A Buyer's Guide to Conic Fitting. In *Proceedings of the British Machine Vision Conference 1995*; British Machine Vision Association: Durham, UK, 1995; pp. 51.1–51.10.
20. Marchand, E.; Uchiyama, H.; Spindler, F. Pose Estimation for Augmented Reality: A Hands-On Survey. *IEEE Trans. Vis. Comput. Graph.* **2016**, *22*, 2633–2651. [CrossRef] [PubMed]
21. Park, F.C.; Martin, B.J. Robot sensor calibration: Solving AX=XB on the Euclidean group. *IEEE Trans. Robot. Autom.* **1994**, *10*, 717–721. [CrossRef]
22. Strobl, K.H.; Hirzinger, G. Optimal Hand-Eye Calibration. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 4647–4653.
23. Jianfei, M.; Qing, T.; Ronghua, L. A Direct Linear Solution with Jacobian Optimization to AX=XB for Hand-Eye Calibration. *WSEAS Trans. Syst. Control* **2010**, *5*, 509–518.
24. Bai, Y.; Zhuang, H. Modeless robots calibration in 3D workspace with an on-line fuzzy interpolation technique. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583), The Hague, The Netherlands, 10–13 October 2004; Volume 6, pp. 5233–5239.
25. Wang, D.; Bai, Y.; Zhao, J. Robot manipulator calibration using neural network and a camera-based measurement system. *Trans. Inst. Meas. Control* **2012**, *34*, 105–121. [CrossRef]
26. Yang, S.; Zhang, C.; Wu, W. Binary output layer of feedforward neural networks for solving multi-class classification problems. *arXiv* **2018**, arXiv:1801.07599. [CrossRef]
27. Liu, B.; Zhang, F.; Qu, X. A Method for Improving the Pose Accuracy of a Robot Manipulator Based on Multi-Sensor Combined Measurement and Data Fusion. *Sensors* **2015**, *15*, 7933–7952. [CrossRef] [PubMed]

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).