

Article

A Deep Reinforcement Learning Enhanced Snow Geese Optimizer for Robot Calibration

Jian Liu ^{1,2} , Yonghong Deng ^{2,3,*} , Canjun Xiao ^{1,2} and Zhibin Li ^{2,3,4,5} 

¹ School of Computer Engineering, Chengdu Technological University, Chengdu 611730, China; liujian@cdtu.edu.cn (J.L.); canjunxiao@cdtu.edu.cn (C.X.)

² Sichuan Provincial Promotion Center of Digital Transformation, Chengdu 611730, China; lizhibin111@outlook.com

³ Dazhou Key Laboratory of Government Data Security, Sichuan University of Arts and Science, Dazhou 635000, China

⁴ School of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China

⁵ Xinjiang Technical Institute of Physics & Chemistry, Chinese Academy of Sciences, Urumqi 830011, China

* Correspondence: dengyhcd@163.com

Abstract: Accurate absolute positioning is essential for industrial robot arms, especially in high-precision manufacturing tasks. Traditional calibration methods often rely heavily on domain-specific knowledge and handcrafted algorithms, making it challenging for broader adoption across disciplines. To tackle this problem, this paper proposes a novel calibration framework based on an enhanced metaheuristic approach named RLPGA, which integrates deep reinforcement learning with the Snow Geese Algorithm (SGA). Unlike conventional strategies where the movement of agents is fully determined by predefined equations, the proposed method leverages a deep policy network to guide individual geese's migration behavior. This network generates adaptive decisions regarding position updates, convergence direction, and flight mode selection. The learned policy enables more flexible and efficient exploration of the calibration parameter space. Experimental results on robot arm calibration tasks demonstrate that RLPGA achieves superior calibration accuracy and robustness compared to existing optimization-based methods, validating its effectiveness and potential for real-world applications.

Keywords: robot calibration; snow geese algorithm; deep reinforcement learning; meta-heuristic optimization; absolute positioning accuracy



Academic Editors: Raouf Fareh and Mohammad H. Rahman

Received: 5 April 2025

Revised: 28 April 2025

Accepted: 3 May 2025

Published: 5 May 2025

Citation: Liu, J.; Deng, Y.; Xiao, C.; Li, Z. A Deep Reinforcement Learning Enhanced Snow Geese Optimizer for Robot Calibration. *Processes* **2025**, *13*, 1407. <https://doi.org/10.3390/pr13051407>

Copyright: © 2025 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 (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In robotic manufacturing systems, the absolute positioning accuracy of industrial robots plays a critical role in the implementation of offline programming. However, due to inherent kinematic errors in the robot body, discrepancies often arise between the theoretical kinematic model and the actual robot behavior [1]. These errors may result from various sources, such as geometric machining inaccuracies during production, misalignments during assembly, and component wear or deformation over extended periods of operation [2]. Collectively, these factors contribute to positional deviations during task execution, thereby compromising the overall precision and stability of the robotic system. To improve the positioning accuracy of industrial robots in real-world working environments, kinematic parameter calibration has emerged as a key approach [3]. This process typically involves four main steps: (1) establishing a mathematical model of the robot's kinematics; (2) acquiring actual end-effector pose data using sensors or external measurement systems; (3) identifying parameter errors, often by applying optimization algorithms to estimate

a parameter set that best matches the observed kinematic; and (4) compensating for the identified model errors to enhance the accuracy of the kinematic model [4].

Based on the established error model, parameter identification typically involves constructing a multi-parameter optimization objective function, which can be solved using either traditional optimization methods or intelligent optimization algorithms [5]. Conventional approaches, such as the Least Squares Method [6], Levenberg–Marquardt (LM) algorithm [7], extended Kalman filter (EKF) [8], and maximum likelihood estimation [9], are often affected by the singularity of the Jacobian matrix and the sensitivity to initial values [10]. Moreover, their performance deteriorates significantly when dealing with nonlinear, high-dimensional, and ill-conditioned calibration problems, which are common in industrial robot applications. In contrast, intelligent optimization algorithms offer effective solutions to these limitations. In recent years, an increasing number of researchers have employed such algorithms for kinematic parameter calibration of industrial robots, demonstrating promising results. Khanesar et al. [11] used intelligent swarm algorithms, such as the artificial bee colony, to calibrate industrial robot DH parameters, improving forward kinematics accuracy by 20.3%. Bastl et al. [12] proposed a robot calibration method using a multi-objective deep-learning evolutionary algorithm optimized by cRVEA, demonstrating improved robustness to measurement noise. Chen et al. [13] developed an improved beetle swarm optimization algorithm with a preference random substitution strategy for kinematic calibration of industrial robots, enhancing positioning accuracy by over 60% compared to traditional methods. Chen et al. [14] utilized a hybrid evolutionary scheme (HOEs) combining multiple evolutionary computing algorithms with a sequential ensemble strategy, memory mechanism, and punishment system, achieving superior performance in industrial robot kinematic calibration. Fan et al. [15] adopted a combined Levenberg–Marquardt and beetle antennae search algorithm to calibrate kinematic parameter errors, significantly enhancing the positioning accuracy of industrial robots. Each intelligent optimization algorithm employs its own unique optimization mechanism. However, existing methods still face limitations such as slow convergence and vulnerability to local optima. Therefore, it is necessary to propose an improved intelligent optimization algorithm to further enhance the accuracy and robustness of industrial robot kinematic parameter calibration.

In 2024, Tian et al. [16] proposed the snow geese algorithm (SGA), demonstrating superior performance over classical algorithms such as particle swarm optimization (PSO) and differential evolution (DE) in terms of solution accuracy and convergence speed for benchmark optimization problems, and successfully applied it to practical engineering problems including tubular column design, piston lever optimization, reinforced concrete beam design, and car side impact design. However, when applied to the kinematic calibration of industrial robots, the Snow Geese Algorithm (SGA) suffers from a significant imbalance between global exploration and local exploitation capabilities. This imbalance leads to a rapid reduction in the effective search space, causing premature convergence and trapping the kinematic parameter estimation in local optima. As a result, the calibration accuracy and robustness are severely compromised, highlighting the need for enhanced optimization strategies.

Although intelligent optimization algorithms have significantly improved the calibration performance of industrial robots, several key challenges remain unresolved. First, most existing methods rely on fixed heuristic strategies for agent movement, limiting their adaptability when facing high-dimensional, nonlinear, and dynamically changing calibration tasks. Second, traditional algorithms often exhibit an imbalance between global exploration and local exploitation, leading to premature convergence and suboptimal calibration accuracy. Third, while emerging algorithms like SGA have demonstrated com-

petitive performance, they still suffer from the rapid shrinkage of search space during optimization, causing the solutions to fall into local optima when applied to complex calibration problems.

Motivated by these challenges, this paper introduces a deep reinforcement learning-enhanced snow geese optimizer (RLSGA) for robot calibration. The main novelty of this research lies in integrating a deep policy network into the Snow Geese Algorithm framework to enable adaptive migration strategies for robot calibration. Key innovations include:

(a) adaptive policy models using deep neural networks to replace traditional heuristic velocity updates, enhancing global search capability and robustness;

(b) a “behavior–learning–perturbation” update mechanism integrating SGA migration behaviors with policy network predictions. This approach significantly improves convergence speed, stability, and accuracy, making RLSGA ideal for calibrating robots with complex or nonlinear error models.

Section 2 describes the kinematic and error modeling methods. Section 3 presents the proposed identification strategy. Experimental results are discussed in Section 4, followed by concluding remarks in Section 5.

2. Kinematic and Error Model

As illustrated in Figure 1, the kinematic structure of an ABB IRB120 robot is developed using the Denavit–Hartenberg (DH) method, a standard approach widely utilized in industrial robot modeling. The figure further illustrates the transformation matrix linking the robot’s base frame to its end-effector frame. The measurement system consists of an ABB IRB120 robot, a nylon cable, and a draw-wire encoder. The calibration procedure focuses on kinematic parameters within the homogeneous transformation matrix that connects the base coordinate system with the end-effector coordinate system. Here, P_i denotes the measurement points ($i = 1, 2, \dots, n$), and P_W indicates the draw-wire anchor point. By positioning the robot’s end-effector at various P_i points, different lengths of the draw wire are obtained. The positioning error is the measured distance between each point P_i and the anchor point P_W . Consequently, the calibration of the end position transforms the problem of reducing positioning errors into minimizing discrepancies in draw-wire length measurements.

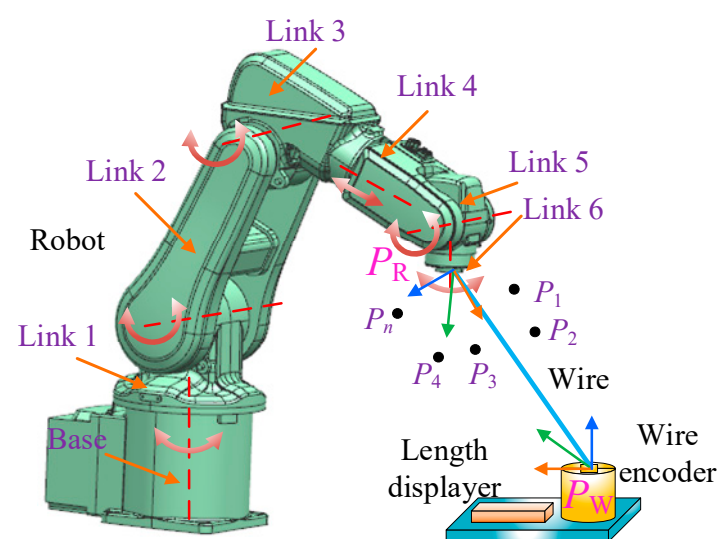


Figure 1. Experimental instruments.

The DH parameters of the robot are summarized in Table 1. The homogeneous transformation matrix that maps coordinates from link $i - 1$ to link i is provided as follows:

$${}^i T_{i-1} = Rot_{z_{i-1}}(\theta_i) Trans_{z_{i-1}}(d_i) Trans_{x_i}(a_i) Rot_{x_i}(\alpha_i) \\ = \begin{bmatrix} c\theta_i & -s\theta_i \cdot c\alpha_i & s\theta_i \cdot s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where i is the index of the joint angle, $i = 1, 2, 3, 4, 5, 6$. θ_i represents the joint angle; d_i denotes the link offset. a_i is the link length. α_i indicates the link twist angle; c stands for the cosine function (cos). s stands for the sine function (sin).

Table 1. Nominal kinematic parameters of the ABB IRB120 robot.

Joint i	α_i /Degree	a_i /mm	d_i /mm	θ_i /Degree
1	−90	0	290	0
2	0	270	0	−90
3	−90	70	0	0
4	90	0	302	0
5	−90	0	0	0
6	0	0	72	0

The nominal end-effector pose matrix T of the robot can be expressed as:

$${}^0 T^6 = {}^0 T_1 T_2 T_3 T_4 T_5 T_6 = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \quad (2)$$

where R and P denote the nominal orientation matrix and the nominal position matrix, respectively. $R \in \mathbb{R}^{3 \times 3}$, $P \in \mathbb{R}^{3 \times 1}$.

θ , d , a , and α are related to the end-effector pose and have an influence on the robot's end-effector position and orientation. Assuming that the four parameters contain errors denoted as Δa_i , Δd_i , $\Delta \alpha_i$, and $\Delta \theta_i$, the actual end-effector pose of the robot can be represented as T^* :

$${}^0 T_6^* = {}^0 T_1^* T_2^* T_3^* T_4^* T_5^* T_6^* \quad (3)$$

The pose variation matrix of the robot end-effector, denoted as dT , can be expressed as:

$$dT = T^* - T = \omega \cdot T \quad (4)$$

where ω is the differential transformation matrix relative to the base frame.

$$\omega = \begin{bmatrix} \delta & d \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -\delta z & \delta y & dx \\ \delta z & 0 & -\delta x & dy \\ -\delta y & \delta x & 0 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

where δ denotes the deviation in the rotation matrix; $d = [dx, dy, dz]^T$ is the differential translation vector; δx , δy , and δz correspond to the orientation errors between the robot's actual and nominal pose.

Substituting Equations (2) into (4) and rearranging yields:

$$\Delta P = \delta \cdot P + d = J \Delta D \quad (6)$$

where ΔP represents the position error between the robot's actual position and its nominal position. J is the Jacobian matrix and ΔD denotes the kinematic error vector; they are respectively expressed as:

$$J = \left[\frac{\partial^0 T_6}{\partial \alpha}, \frac{\partial^0 T_6}{\partial a}, \frac{\partial^0 T_6}{\partial d}, \frac{\partial^0 T_6}{\partial \theta} \right] \quad (7)$$

$$\Delta D = [\Delta \alpha, \Delta a, \Delta d, \Delta \theta]^T \quad (8)$$

Hence, the fundamental objective of robot geometric parameter calibration is to identify the discrepancies between the nominal and actual end-effector poses under various joint configurations $(\theta_{1j}, \theta_{2j}, \theta_{3j}, \theta_{4j}, \theta_{5j}, \theta_{6j})$, and to determine the optimal set of geometric parameter deviations $(\Delta a_i, \Delta d_i, \Delta \alpha_i, \Delta \theta_i)$ by means of an optimization algorithm, thereby minimizing the pose error.

When using a cable encoder for robot calibration, orientation errors are neglected, and only positional deviations are considered. Let L denote the measured cable length and L^* represent the theoretical length computed based on the kinematic model. In accordance with the calibration principles, a corresponding loss function can be constructed to optimize the D-H parameters.

$$f(x) = \min \frac{1}{N} \sum_{i=1}^N \|L_i - L_i^*\|_2 \quad (9)$$

where N is the number of measurement samples, and the expression for L_i is provided as follows:

$$L_i = \|P_i - P_W\|_2 \quad (10)$$

where P_W represents the fixed position of the cable encoder, which is directly acquired through the measurement software.

Accordingly, based on the formulated objective function, the problem can be characterized as a typical complex, high-dimensional, and nonlinear equation-solving task.

3. Kinematic Parameter Error Identification

3.1. SGA Algorithm for Identification

The SGA algorithm is primarily divided into three phases: the Initialization Phase, the Exploration Phase, and the Exploitation Phase.

In the Initialization Phase, the positions of individuals are initialized. Candidate solutions X for kinematic parameter errors are randomly generated within the search space. The initial positions are determined based on the population size, the boundaries of the solution space, and the problem dimension. The initialization formula is expressed as follows:

$$X = lb + rand \times (ub - lb) \quad (11)$$

During the initialization of the SGA optimization process, the deviations in link lengths and offsets (a_i and d_i) were constrained within ± 8 mm, and the deviations in joint angles and twist angles (θ_i and α_i) were constrained within $\pm 8^\circ$, based on the mechanical design tolerances of the ABB IRB120 robot. This ensured that the optimization process remained within physically meaningful bounds and prevented parameter drift due to redundant parameterizations.

Exploration phase-velocity update: In the exploration phase, individuals update their velocities to search for promising regions in the solution space. The velocity-update mechanism enables the algorithm to explore diverse areas and avoid premature convergence. The

velocity of each individual is typically influenced by its current state, the positions of other individuals, and specific control parameters. The velocity update formula is provided as:

$$V^{t+1} = c \cdot V^t + a \quad (12)$$

where V^t and V^{t+1} denote the velocities at the t -th and $(t + 1)$ -th iterations, respectively. Here, c is the weighting factor, and a represents the acceleration coefficient. They are calculated as follows:

$$c = \frac{4t}{Te^{4t/T}} \quad (13)$$

$$a = (x_b^t - x_i^t) - 1.29(v_i^t)^2 \cdot \sin(\theta) \quad (14)$$

where T is the maximum iteration number, x_b^t represents the global best solution at the t -th iteration, x_i^t denotes the i -th candidate solution at the t -th iteration, and θ refers to the fight angle.

Position update (exploration phase): In the exploration phase, individuals update their positions based on their current velocities. This update guides individuals to move through the search space and discover new potential solutions. The position update formula is typically expressed as:

$$x_i^{t+1} = x_i^t + b(x_b^t - x_i^t) + v_i^{t+1} \quad (15)$$

where b is the weighting factor.

Exploitation Phase: In the exploitation phase, the algorithm simulates the “linear” flight pattern of snow geese to perform intensive local search around promising regions. This strategy enhances the exploitation capability of the algorithm by focusing on refining the current best solutions. The position update in this phase is governed by the following formula:

$$x_i^{t+1} = \begin{cases} x_i^t + (x_i^t - x_b^t) \cdot \gamma, \gamma > 0.5 \\ x_b^t + (x_i^t - x_b^t) \cdot \gamma \oplus \text{Brownian}(d), \gamma \leq 0.5 \end{cases} \quad (16)$$

where γ is a random number within the range $[0, 1]$, $\oplus$ denotes element-wise multiplication, and $\text{Brownian}(d)$ represents Brownian motion in d -dimensional space.

3.2. Enhancing Snow Geese Algorithm with Deep Neural Networks

To further improve the identification accuracy of the SGA and to avoid premature convergence to local optima, a policy network $\pi_\theta(\cdot)$ is introduced to replace the traditional velocity update formula of the standard SGA. The network takes as input the current position of the individual, the current velocity, the current global best position, and the current global center position. It outputs the updated velocity for the next iteration. As shown in Figure 2, a fully connected neural network is designed to capture the nonlinear relationship between the input features and the updated velocity. The neural network structure (4-30-30-30-30-1) comprises five hidden layers, offering sufficient depth to model complex dynamics during the optimization process.

The state vector is constructed for each individual, representing its status at the t -th iteration. The state vector for an individual at iteration t is defined as:

$$s_i^t = [x_i^t, v_i^t, x_b^t, x_c^t] \in R^{4D} \quad (17)$$

where:

x_i^t represents the current estimated kinematic parameter error of the individual,

v_i^t denotes the current adjustment velocity, corresponding to the step size and direction of parameter correction,

x_b^t refers to the current global best estimated kinematic parameter error among all individuals, serving as the best-known calibration solution so far,

x_c^t is the current global center position of the population, reflecting the collective search status.

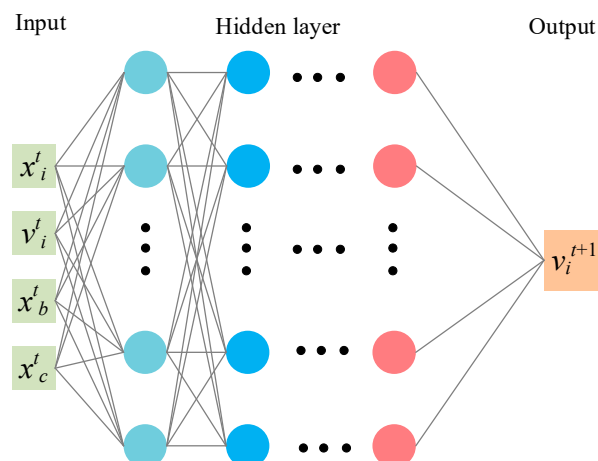


Figure 2. Network architecture for SGA. Note that the network structure is (4-30-30-30-30-30-1).

The policy network takes this state vector as input and outputs a new adjustment velocity v_{i+1}^i . This predicted velocity represents the next movement direction and step size for updating the estimated kinematic parameters, aiming to minimize the overall calibration residuals. The policy network is defined as:

$$v_i^{t+1} = \pi_\theta(s_i^t) \quad (18)$$

The output v_i^{t+1} corresponds to the velocity update for the next iteration.

The training objective of the policy network is formulated as a supervised learning task. Training samples (S_i^t, v_i^{t+1}) are collected by observing the actual behavior of the standard SGA. The loss function is defined as:

$$\ell(\theta) = \frac{1}{M} \sum_{i=1}^M \left\| \pi_\theta(S_i^t - v_i^{t+1}) \right\| \quad (19)$$

where M denotes the number of training samples.

In the context of robot calibration, the input state vector elements correspond to the estimated kinematic parameter deviations and their dynamic adjustment behaviors, while the output velocity adjustment guides the refinement of the calibration parameter estimates to achieve higher positioning accuracy.

The position update is performed using the updated velocity according to the following formula:

$$x_i^{t+1} = x_i^t + v_i^{t+1} + \Delta_{mig}^t \quad (20)$$

where Δ_{mig}^t is a perturbation term derived from the modeling of snow geese migration behavior, including V-formation, linear flight, and Brownian motion.

$$\Delta_{mig}^t = \begin{cases} a(x_b^t - x_i^t) + b(x_c^t - x_i^t), & \theta < \pi \\ (x_i^t - x_b^t) \cdot \gamma, & \gamma > 0.5 \\ (x_i^t - x_b^t) \cdot \gamma \oplus \text{Brownian}(\text{dim}), & \gamma \leq 0.5 \end{cases} \quad (21)$$

A convergence-checking mechanism was incorporated into the RLPGA implementation, where convergence is confirmed based on objective function improvement thresholds, and a convergence alarm is triggered if the convergence condition is not satisfied within the maximum allowable iterations.

In each iteration, the state vector of each agent is processed by the policy network to infer the new adjustment velocity, which is subsequently combined with SGA migration updates for position refinement. The updated position is then evaluated for calibration accuracy, completing one cycle of interaction between the learning and optimization components.

Algorithm 1 presents the pseudo-code of the proposed RLPGA algorithm for solving the robot calibration problem. RLPGA builds upon the conventional SGA by integrating a deep policy network to model individual migration behaviors adaptively. Instead of relying on fixed-velocity update formulas, the algorithm leverages learned state-action mappings to predict individual movement directions, thereby enhancing its adaptability and convergence efficiency in high-dimensional and nonlinear optimization scenarios. Meanwhile, the inherent biological behaviors of SGA—namely, the V-formation and straight-line flight—are preserved and combined with the learned strategy to form a collaborative search mechanism. This hybrid approach makes SGA-RL particularly suitable for complex robot calibration tasks involving multi-source uncertainties and nonlinear parameter dependencies.

Unlike conventional gradient-based methods that require explicit Jacobian and Hessian computations, the proposed RLPGA operates through population-based stochastic updates guided by a deep policy network, thereby avoiding numerical instabilities associated with near-singular configurations and heterogeneous constraint conditions.

3.3. Performance Evaluation of RLPGA

To validate the performance improvement introduced by RLPGA, several representative benchmark functions (f1, f3, f5, f9, f13, f15, f16, f20, f22) from the CEC2005 test suite are selected for evaluation. The original SGA is used as a baseline for comparison. Figure 3 presents the experimental results, illustrating the convergence behavior of both algorithms across the selected test functions. As shown in Figure 3, the improved RLPGA demonstrates significantly better performance than the original SGA in terms of both convergence speed and solution accuracy. Specifically, RLPGA converges faster and reaches lower final objective values on most functions, indicating its superior optimization capability. These results confirm the effectiveness of incorporating deep reinforcement learning into the optimization process. The learning-guided strategy enables RLPGA to dynamically adapt its search direction, enhancing global exploration and local exploitation abilities in complex, high-dimensional search spaces.

Algorithm 1 The pseudocode of the RLSGA algorithm**RLSGA Algorithm****Input:** function f , $\{q_1, q_2, q_3, q_4, q_5, q_6\}$, $\{L^*_1, L^*_2, \dots, L^*_n\}$ **/--Note: Initialization--/**

```

1.      Initialize: Search space bounds [Xmin, Xmax]
2.      Initialize: Maximum iteration number  $T$ , Population size  $M$ 
3.      Pre-trained policy network  $\pi_\theta(s)$  // or online training strategy
4.      Initialize population position matrix  $X \in R^{M \times d}$  and matrix  $V \in R^{M \times d}$ 
         $X^0 \sim U(Xmin, Xmax)$ ,  $V^0 \leftarrow 0$ 
5.      Evaluate initial fitness values:  $FV_i = f(x_i^0)$ 
6.      Set global best:  $x_b \leftarrow \text{argmin}(f(X))$ 
7.       $t = 0$ 
8.      while  $t < T$  do
9.           $\theta = 2\pi \cdot t/T$  // angle for mode selection
10.         Compute population center  $x_c = \sum (x_i \cdot f(x_i)) / \sum f(x_i)$ 
11.         for  $i = 1$  to  $M$ 
12.             Construct state vector:  $S_i^t = [x_i^t, v_i^t, x_{b1}^t, x_c^t]$ 
13.             Predict new velocity by strategy:  $v_i^{t+1} = \pi_\theta(S_i^t)$ 
14.         end for
15.         if  $\theta < \pi$  then // Herringbone (exploration) mode
16.             Compute random coefficients:
17.                  $b = 4 \cdot \text{rand}() - 2$ ,  $d = 3 \cdot \text{rand}() - 1.5$ ,  $e = 2 \cdot \text{rand}() - 1$ 
18.             Sort individuals by fitness
19.             for  $i = 1$  to  $M$ 
20.                 if  $i \leq M/5$ 
21.                      $x_i^{t+1} = x_i^t + b \cdot (x_c^t - x_i^t) + v_i^{t+1}$ 
22.                 else if  $i > M/5$  and  $i \leq 4M/5$ 
23.                      $x_i^{t+1} = x_i^t + b \cdot (x_c^t - x_i^t) + d \cdot (x_c^t - x_i^t) + v_i^{t+1}$ 
24.                 else
25.                      $x_i^{t+1} = x_i^t + b \cdot (x_c^t - x_i^t) + d \cdot (x_c^t - x_i^t) - e \cdot (x_i^t - x_i^t) + v_i^{t+1}$ 
26.                 end if
27.             end for
28.         else // Straight-line (exploitation) mode
29.             for  $i = 1$  to  $M$ 
30.                  $\gamma = \text{rand}()$ 
31.                 if  $\gamma > 0.5$ 
32.                      $x_i^{t+1} = x_i^t + (x_i^t - x_b^t) \cdot \gamma$ 
33.                 else
34.                      $x_i^{t+1} = x_b^t + (x_i^t - x_b^t) \cdot \gamma \oplus \text{Brownian}(\text{dim})$ 
35.                 end if
36.             end for
37.         end if
38.         Apply boundary control to  $x_i^{t+1}$ 
39.         Re-evaluate fitness values:  $\text{Fit}_i = f(x_i^{t+1})$ 
40.         Update  $x_b^t$  if better solution is found
41.          $t = t + 1$ 
42.     end while

```

/--Note: Training Ends--/**Output:** x_{best}

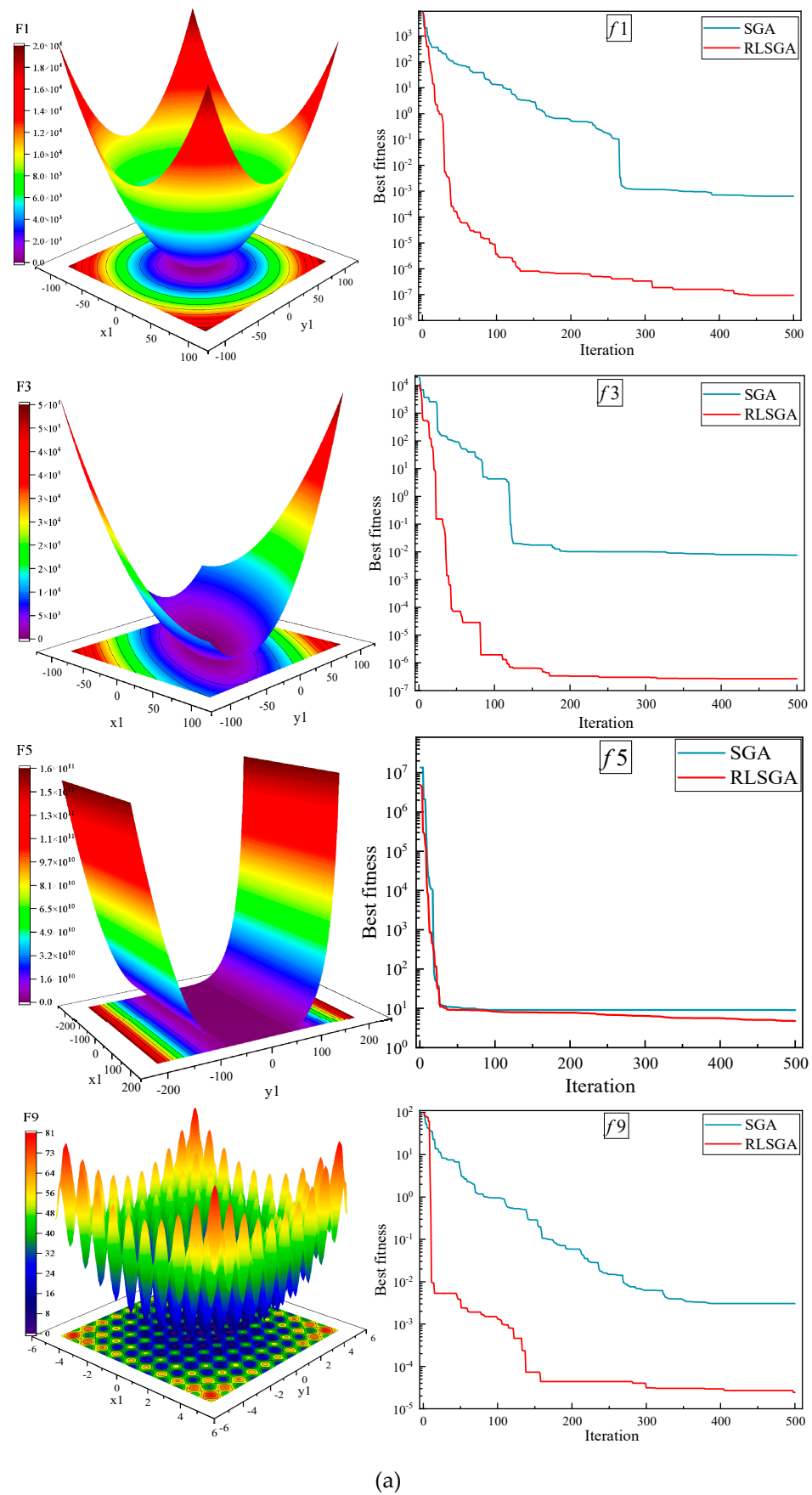


Figure 3. Cont.

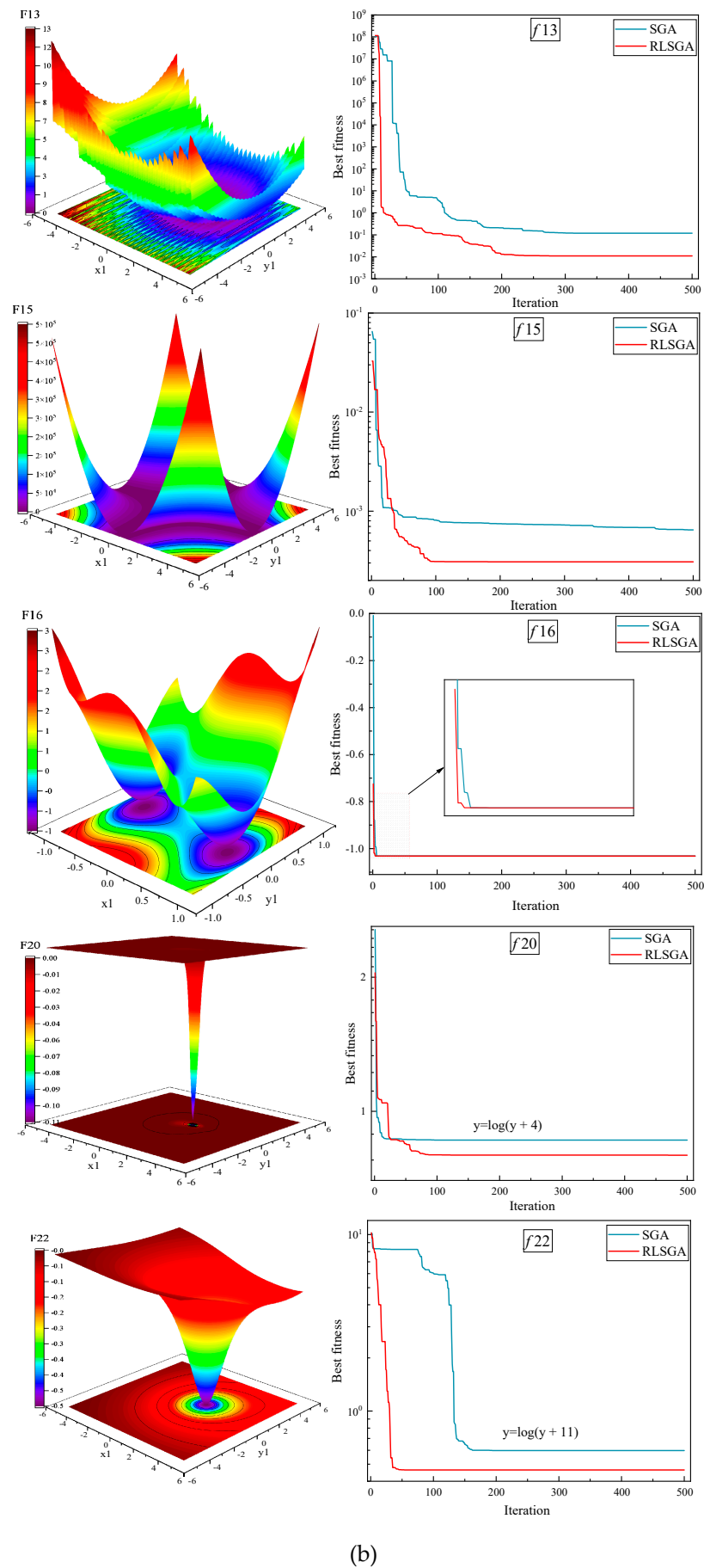


Figure 3. Performance comparison of RLSGA and SGA on CEC2025 benchmark functions. (a) f1, f3, f5, f9; (b) f13, f15, f16, f20, f22.

4. Robot Kinematic Calibration Experiment

4.1. Experimental Conditions

4.1.1. Experimental Platform and Data Acquisition

An experimental platform, depicted in Figure 4, is constructed to enable high-precision calibration of the robot. The setup consists of an ABB IRB120 industrial robotic arm, featuring a repeatability of 0.01 mm, along with a cable-type displacement sensor, a digital displacement display unit, and an RS485 communication interface. This configuration ensures accurate displacement acquisition and stable data transmission during the calibration process. Table 2 provides detailed specifications of the drawstring displacement sensor.

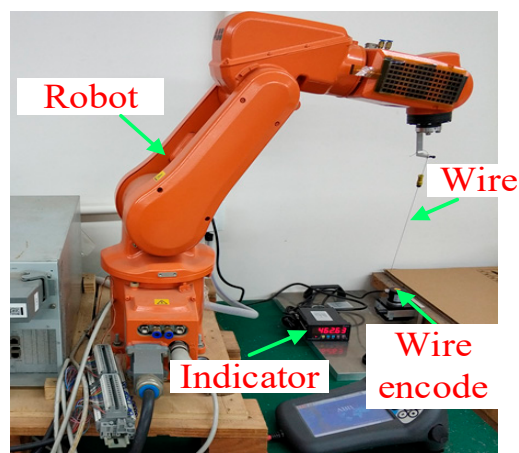


Figure 4. Robot calibration devices.

Table 2. Specifications of the drawstring displacement sensor.

Parameter	Value
Measuring range	2000 mm
Maximum operating speed	1000 m/s
Retraction force	5 N
Measurement resolution	0.004 mm
Operating temperature range	−25 °C~+85 °C

To ensure the accuracy and representativeness of the calibration dataset, measurement points were comprehensively distributed across the entire operational workspace of the ABB IRB120 industrial robot. A total of 1042 measurement positions were strategically selected, and their spatial coordinates were captured using a high-resolution drawstring (cable-type) displacement sensor with a resolution of 0.004 mm. To facilitate efficient and consistent data recording, a customized acquisition program was developed using LabVIEW software, enabling the automated logging of sample information. To mitigate potential ill-posedness caused by degenerate configurations, the measurement points were carefully distributed across the entire reachable workspace of the robot, covering diverse joint configurations and spatial locations. This ensures sufficient excitation of all degrees of freedom during calibration.

From the collected measurement data, 200 representative samples were randomly selected under a uniform distribution to construct each calibration case. To mitigate the potential effects of sampling bias and improve statistical robustness, the random selection process was repeated 10 times, resulting in 10 distinct calibration data cases. In each case, the selected dataset was divided into two subsets: 160 samples (80%) were allocated for training purposes to fit the calibration models, while the remaining 40 samples

(20%) were reserved for independent testing to evaluate generalization capability. This consistent partitioning strategy facilitates a balanced assessment of both model accuracy and robustness across different data scenarios. During model evaluation, each method's calibration performance was assessed separately on these 10 data cases. The final reported results, including RMSE, Std, and Max positioning errors, represent the mean and standard deviation across all 10 cases, thereby ensuring that the evaluation is statistically reliable and not dependent on a specific sampling instance.

Furthermore, to support transparency and encourage reproducibility, the resulting dataset, along with detailed documentation, has been made publicly available at <https://github.com/Lizhibing1490183152/RobotCali> (accessed on 1 August 2022), providing a reliable benchmark for further research in robot calibration studies.

4.1.2. Evaluation Metrics and Comparison Methods

In the context of large-scale industrial applications, a primary objective of robot calibration is to enhance the robot's absolute positioning accuracy, thereby meeting the stringent demands of high-precision manufacturing processes [7–10]. To comprehensively assess the effectiveness of the calibration method, three widely recognized statistical indicators are employed as performance metrics: root mean squared error (RMSE), standard deviation (Std), and maximum error (Max). These metrics provide insights into the overall positioning precision, consistency, and worst-case deviations of the robotic system. The mathematical definitions of these evaluation indicators are formulated as follows:

$$\begin{aligned} Max &= \max \left\{ \sqrt{(L_i - L_i^*)^2} \right\}, i = 1, 2, \dots, m \\ Std &= \frac{1}{m} \sum_{i=1}^m \sqrt{(L_i - L_i^*)^2} \\ RMSE &= \sqrt{\frac{1}{m} \sum_{i=1}^m (L_i - L_i^*)^2} \end{aligned} \quad (22)$$

To validate the effectiveness and superiority of the proposed calibration approach, a series of comparative experiments were conducted against several representative state-of-the-art methods. These benchmark techniques are selected based on their relevance and proven performance in the field of robot kinematic calibration. The specific comparative strategies and experimental settings are outlined as follows, aiming to ensure a fair and comprehensive evaluation of calibration accuracy across different algorithms.

M1: The extended Kalman filter (EKF) algorithm, as introduced in [8], is capable of effectively handling non-Gaussian noise within the calibration framework.

M2: An LM-based calibration model [7], widely used in practice, optimizes the calibration objective by minimizing the residual gradient to identify D-H parameters.

M3: The particle swarm optimization (PSO) algorithm [17] mimics the flocking behavior of birds to efficiently search for optimal solutions.

M4: The differential evolution (DE) algorithm [18] is a population-based optimization method capable of performing accurate multi-objective robot calibration.

M5: The beetle antennae search (BAS) algorithm [13] emulates the foraging mechanism of beetles to solve complex optimization tasks.

M6: The artificial bee colony (ABC) [11] algorithm is employed for calibrating the D-H parameters of industrial robots.

M7: The snow geese algorithm (SGA) [16,19], inspired by the flocking behavior of snow geese, is utilized for calibrating the robot's kinematic parameters.

M8: The proposed method, a deep reinforcement learning-enhanced snow geese algorithm (RLSGA), is applied to robot calibration to further improve the accuracy of parameter identification.

Table 3 presents the hyperparameter configurations that yielded the best calibration performance for each model during the tuning process. Each setting was determined based on extensive empirical evaluation, aiming to maximize the accuracy and robustness of the respective optimization strategies.

Table 3. Hyper-parameter settings.

Method	Hyper Parameter
M1	$P_k = 0.5 \times 10^{-7}, Q_k = 0.5 \times 10^{-7}, R_k = 0.2 \times 10^{-4}$
M2	
M3	$\mu = 0.5 \times 10^{-3}$
M4	$M = 100, c_1 = 1.2, c_2 = 1.2, \omega = 0.8$
M5	$M = 100, F = 0.5, CR = 0.1$
M6	$d_0 = 0.01, \delta_0 = 0.01, \text{eta} = 0.95$
M7	$M = 100, Eb = 50, Ob = 25, \text{limit} = 1000$
M8	$M = 100, L = 10, \alpha = 0.2, \beta = 0.05$
	$lr = 0.005, \gamma = 0.9, \varepsilon = 0.9,$ network structure (4-30-30-30-30-1)

4.2. Experimental Results

Figures 5–7 present, respectively, the calibration effectiveness, the positioning accuracy, and the convergence trends achieved by models M1 through M8. Table 4 summarizes the comparative calibration results, including the overall time consumption associated with each evaluated method. Furthermore, the optimized D-H parameter compensation values obtained through M8, namely, the proposed RLSGA approach, are detailed in Table 5.

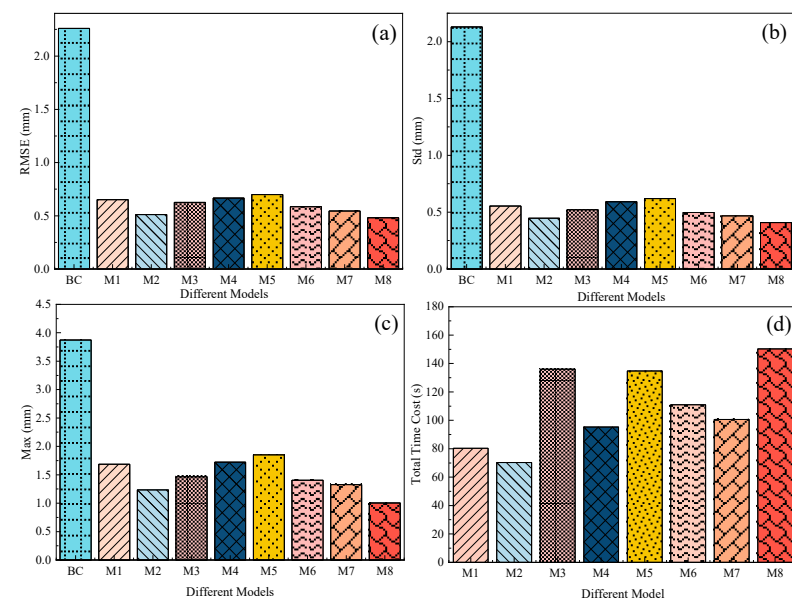


Figure 5. Accuracy and total time cost comparison of calibration methods. (a) RMSE; (b) Std.; (c) Max; (d) Time cost.

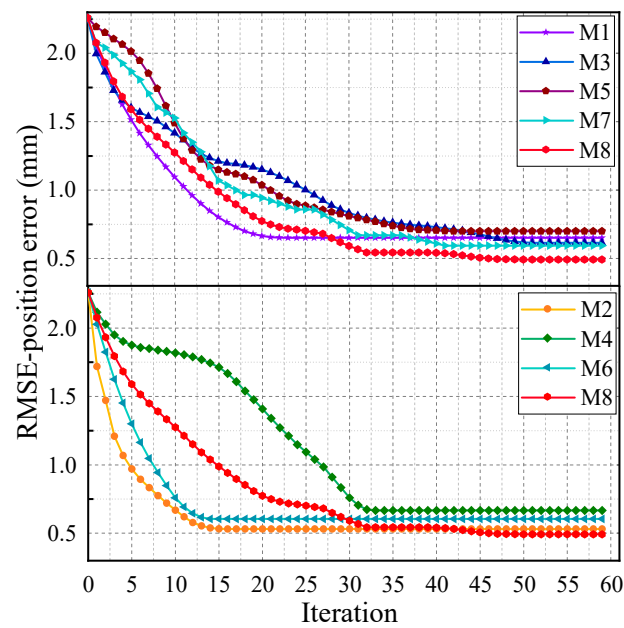


Figure 6. Training curves of different calibration methods.

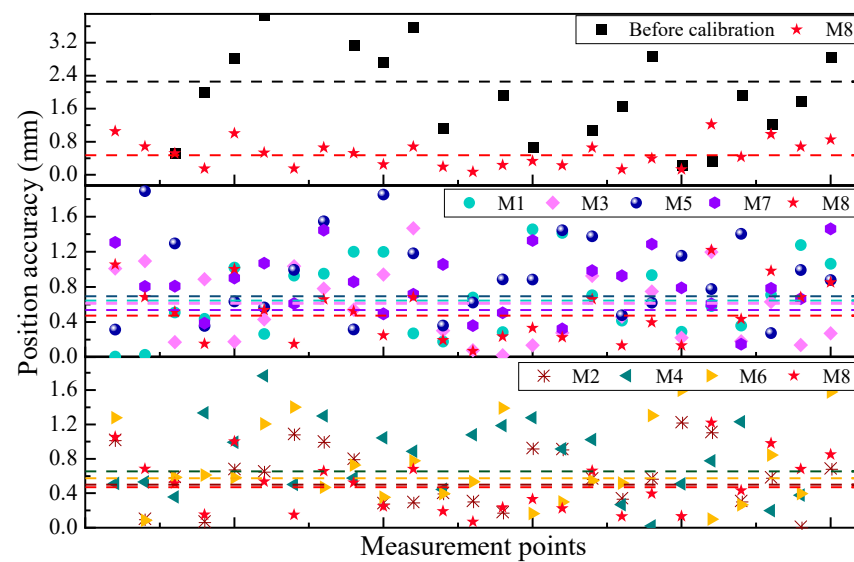


Figure 7. Calibrated position error using different methods.

Table 4. Comparison of calibration results.

Item	RMSE (mm)	STD (mm)	Max (mm)	Time (s)
Before	2.263	2.131	3.876	/
M1	0.652	0.553	1.685	80.321
M2	0.511	0.446	1.235	70.348
M3	0.625	0.52	1.467	135.982
M4	0.667	0.592	1.721	95.351
M5	0.701	0.622	1.851	134.761
M6	0.585	0.498	1.401	110.917
M7	0.544	0.467	1.328	100.648
M8	0.481	0.408	1.002	150.327

Table 5. Calibrated kinematic parameters of the ABB IRB120 robot.

Joint i	α_i/Degree	a_i/mm	d_i/mm	θ_i/Degree
1	−90.214	0.175	292.375	2.176
2	0.0347	273.847	0.728	−90.758
3	−91.387	73.695	−0.743	0.016
4	88.367	0.671	302.684	0.748
5	−90.476	0.058	−1.736	−3.753
6	0.176	1.278	72.247	1.274

Based on these experimental outcomes, several noteworthy observations can be made:

The proposed RLPGA algorithm (M8) achieves effective calibration results for the robot system. Table 4 and Figure 5 present the calibration performance before and after applying the optimization strategy. M8 exhibits significant improvements across all evaluation metrics when compared to the uncalibrated condition. Specifically, M8 reduces the RMSE from 2.263 mm to 0.481 mm, the Std from 2.131 mm to 0.408 mm, and the Max from 3.876 mm to 1.002 mm, corresponding to improvements of 78.75%, 80.85%, and 74.15%, respectively.

M8 exhibits the best overall performance in terms of robot calibration accuracy, compared with M1 to M7. As reported in Table 4 and Figure 5, the RMSE, Std, and Max obtained by M8 are 0.481 mm, 0.408 mm, and 1.002 mm, respectively. Taking M2, the most accurate among the baseline methods, as a comparison point, its corresponding RMSE, Std, and Max are 0.511 mm, 0.446 mm, and 1.235 mm, respectively. On this basis, M8 achieves accuracy improvements of 5.87%, 8.52%, and 18.87%, respectively, in the three metrics. These findings confirm that the proposed RLPGA approach delivers notable enhancements in accuracy and is thus effective in improving robot calibration performance.

M8 is an enhanced version of the standard SGA algorithm (M7), incorporating deep reinforcement learning to guide the optimization process. Compared to M7, M8 achieves notable improvements in calibration accuracy. Specifically, as shown in Table 2 and Figure 5, M8 reduces the RMSE from 0.544 mm to 0.481 mm (an 11.58% improvement), the Std from 0.467 mm to 0.408 mm (12.63% improvement), and the Max from 1.328 mm to 1.002 mm, yielding a significant 24.55% reduction. These results demonstrate that integrating a deep policy network into the SGA framework can effectively enhance optimization behavior, leading to better calibration accuracy. The proposed RLPGA outperforms its base version (SGA), verifying the effectiveness of reinforcement learning in guiding population-based evolutionary strategies.

In addition to calibration accuracy, the computational cost is an important factor when evaluating the overall performance of optimization algorithms. As shown in Table 2 and Figure 5, M8 (RLPGA) exhibits the longest execution time of 150.327 s, which is higher than all the baseline methods (M1–M7). This increased time consumption is primarily attributed to the additional training and inference processes involved in the deep reinforcement learning strategy embedded in RLPGA. For example, M8 requires approximately 49.7% more time than M6 (100.648 s) and nearly doubles the runtime compared to M2 (70.348 s), which was the fastest among all tested methods. While M8 achieves the best calibration accuracy across all evaluation metrics, these results also reveal a trade-off between optimization precision and computational efficiency. Nevertheless, such a cost is acceptable and often worthwhile in applications where accuracy is of paramount importance.

Figure 6 illustrates the convergence behavior of different calibration methods in terms of RMSE-position error over 60 iterations. It is evident that M8 (RLPGA) consistently outperforms the other methods in both convergence speed and final accuracy. In the top subfigure, which compares M1, M3, M5, M7, and M8, the M8 curve exhibits a steep decline

in the early iterations and quickly converges to a lower RMSE level than its counterparts. While M1 and M5 plateau at relatively higher error levels, M8 continues to descend and stabilizes with minimal fluctuations. Notably, M7 (the original SGA) converges more slowly and reaches a higher final RMSE than M8, highlighting the effectiveness of introducing deep reinforcement learning in RLPGA. Similarly, in the bottom subfigure comparing M2, M4, M6, and M8, M8 again achieves the fastest and smoothest convergence trajectory. Although M2 and M6 initially decline rapidly, their curves plateau earlier and settle at higher error levels compared to M8. These results demonstrate that M8 not only reaches the lowest final RMSE but also exhibits superior convergence characteristics, validating the advantage of the proposed method in both optimization capability and learning efficiency during calibration.

A total of 200 measurement points were collected using an ABB IRB120 industrial robot. Upon completing the calibration process for these measurement points, the resulting positioning accuracy of each method is evaluated and compared, as presented in Figure 7. Figure 7 illustrates the calibrated position errors of different methods across all measurement points. Each horizontal group of markers corresponds to a specific calibration method, where the distribution of points reflects the residual positioning error for the test samples. It can be observed that before calibration (black squares), the positioning errors are significantly larger and more dispersed, often exceeding 2 mm. After calibration, all methods (M1–M8) substantially reduce the error magnitude. Among them, M8 (the proposed RLPGA method, red stars) exhibits the lowest and most concentrated residual error distribution. The error points of M8 consistently remain below 1.0 mm, with most residuals clustered around 0.4–0.6 mm. In contrast, other methods such as M1, M3, and M5 show wider scatter and larger error magnitudes, indicating less stable calibration performance. Furthermore, the dashed lines representing the RMSE values clearly indicate that M8 achieves the lowest RMSE among all methods. Specifically, the horizontal dashed line for M8 is located lower than those for M1–M7, validating the superior positioning accuracy and consistency of the proposed method. Notably, while M7 (SGA) performs reasonably well, its residual distribution is still broader compared to M8, emphasizing the benefits brought by integrating deep reinforcement learning into the optimization framework. These observations from Figure 7 strongly confirm that the proposed RLPGA method (M8) not only minimizes the overall positioning error but also ensures better consistency across different measurement points, thus demonstrating its effectiveness in improving the absolute positioning accuracy of industrial robots.

5. Conclusions

To enhance the calibration accuracy of industrial robots, this paper proposed a novel RLPGA algorithm, which integrates deep reinforcement learning with a biologically inspired metaheuristic framework. A series of benchmark function tests and robot calibration experiments were conducted to verify the effectiveness and superiority of the proposed method. The main conclusions are summarized as follows:

(1) In comparison with the original SGA, the proposed RLPGA demonstrates significantly improved performance on the CEC2005 benchmark functions in terms of convergence speed and solution accuracy. By introducing a learned policy network to adaptively guide the optimization process, RLPGA effectively overcomes the limitations of premature convergence and local optima, enhancing both global exploration and local exploitation capabilities.

(2) In industrial robot calibration tasks, RLPGA achieves the highest calibration accuracy among eight representative methods, including both classical optimization algorithms (e.g., PSO, DE) and advanced heuristics (e.g., BAS, ABC). Specifically, RLPGA reduces the

RMSE, Std, and Max to 0.481 mm, 0.408 mm, and 1.002 mm, respectively, representing the best performance across all evaluated metrics. Moreover, its convergence curves are smoother and faster than those of the baseline methods, confirming the robustness and learning efficiency of the proposed strategy.

(3) Although RLPGA incurs higher computational cost due to the training and inference procedures of the deep policy network, this trade-off is acceptable in high-precision applications where calibration accuracy is of critical importance. Overall, the proposed RLPGA provides a promising solution for robot calibration tasks with complex, nonlinear error models and offers potential for broader applications in intelligent manufacturing and adaptive robotics.

In future work, several promising research directions can be explored to further enhance the effectiveness and applicability of the proposed RLPGA method. First, developing online learning strategies could enable real-time adaptation of the calibration model during robot operation, improving dynamic accuracy under changing conditions. Second, extending the kinematic error modeling to incorporate thermal effects, payload variations, and other environmental factors could further improve calibration robustness. Third, applying the RLPGA framework to different types of robots, such as collaborative or redundant robots, would validate its generalization capability across various robotic platforms. These potential extensions will be the focus of our subsequent research.

Author Contributions: Conceptualization, J.L.; methodology, J.L.; software, Y.D.; validation, Y.D., and Z.L. formal analysis, J.L. and C.X.; investigation, J.L. and C.X.; resources, Z.L.; data curation, Z.L.; writing—original draft preparation, J.L.; writing—review and editing, Y.D.; visualization, J.L.; supervision, Z.L.; project administration, Z.L.; funding acquisition, Y.D. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by the Natural Science Foundation of Sichuan, China (NO.25LHJJ0373), Dazhou Key Laboratory of Government Data Security (No. ZSAQ202422 and No. ZSAQ202401), and the National Funded Postdoctoral Research Program (No. GZC20241900), Natural Science Foundation Program of Xinjiang Uygur Autonomous Region (No. 2024D01A141), Tianchi Talents Program of Xinjiang Uygur Autonomous Region (Li Zhibin) and Postdoctoral Fund of Xinjiang Uygur Autonomous Region (Li Zhibin).

Data Availability Statement: The data that support the findings of this study are openly available in RobotCali: <https://github.com/Lizhibing1490183152/RobotCali> (accessed on 1 August 2022).

Acknowledgments: The authors would like to express their sincere gratitude to the technical staff and research members of the laboratory team for their assistance in conducting robot calibration experiments and providing valuable support throughout the study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Li, Z.; Li, S.; Luo, X. An overview of calibration technology of industrial robots. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 23–36. [\[CrossRef\]](#)
2. Deng, Y.; Hou, X.; Li, B.; Wang, J.; Zhang, Y. A highly powerful calibration method for robotic smoothing system calibration via using adaptive residual extended Kalman filter. *Robot. Comput.-Integr. Manuf.* **2024**, *86*, 102660. [\[CrossRef\]](#)
3. Dehghani, M.; McKenzie, R.A.; Irani, R.A.; Ahmadi, M. Robot-mounted sensing and local calibration for high-accuracy manufacturing. *Robot. Comput.-Integr. Manuf.* **2023**, *79*, 102429. [\[CrossRef\]](#)
4. Maghami, A.; Imbert, A.; Côté, G.; Monsarrat, B.; Birglen, L.; Khoshdarregi, M. Calibration of multi-robot cooperative systems using deep neural networks. *J. Intell. Robot. Syst.* **2023**, *107*, 55. [\[CrossRef\]](#)
5. Deng, Y.; Hou, X.; Li, B.; Wang, J.; Zhang, Y. A novel method for improving optical component smoothing quality in robotic smoothing systems by compensating path errors. *Opt. Express* **2023**, *31*, 30359–30378. [\[CrossRef\]](#) [\[PubMed\]](#)
6. 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*, 136060–136070. [\[CrossRef\]](#)

7. Feng, A.; Zhou, Y.; Zhang, R.; Zhao, W.; Li, Z.; Zhu, M. A novel kinematic calibration method for robot based on the Levenberg–Marquardt and improved Marine Predators algorithm. *Measurement* **2025**, *243*, 116125. [[CrossRef](#)]
8. Yin, F.; Wang, L.; Tian, W.; Zhang, X. Kinematic calibration of a 5-DOF hybrid machining robot using an extended Kalman filter method. *Precis. Eng.* **2023**, *79*, 86–93. [[CrossRef](#)]
9. Ma, L.; Bazzoli, P.; Sammons, P.M.; Landers, R.G.; Bristow, D.A. Modeling and calibration of high-order joint-dependent kinematic errors for industrial robots. *Robot. Comput.-Integr. Manuf.* **2018**, *50*, 153–167. [[CrossRef](#)]
10. Deng, Y.; Hou, X.; Li, B.; Wang, J.; Zhang, Y. A novel positioning accuracy improvement method for polishing robot based on Levenberg–Marquardt and opposition-based learning squirrel search algorithm. *J. Intell. Robot. Syst.* **2024**, *110*, 8. [[CrossRef](#)]
11. Khanesar, M.A.; Yan, M.; Isa, M.; Piano, S.; Branson, D.T. Precision Denavit–Hartenberg parameter calibration for industrial robots using a laser tracker system and intelligent optimization approaches. *Sensors* **2023**, *23*, 5368. [[CrossRef](#)] [[PubMed](#)]
12. Bastl, P.; Chakraborti, N.; Valášek, M. Evolutionary algorithms in robot calibration. *Mater. Manuf. Process.* **2023**, *38*, 2051–2070. [[CrossRef](#)]
13. Chen, X.; Zhan, Q. The kinematic calibration of an industrial robot with an improved beetle swarm optimization algorithm. *IEEE Robot. Autom. Lett.* **2022**, *7*, 4694–4701. [[CrossRef](#)]
14. Chen, T.; Li, S.; Qiao, Y.; Luo, X. A robust and efficient ensemble of diversified evolutionary computing algorithms for accurate robot calibration. *IEEE Trans. Instrum. Meas.* **2024**, *73*, 7501814. [[CrossRef](#)]
15. Fan, M.; Zhao, H.; Wen, J.; Yu, L.; Xia, H. A novel calibration method for kinematic parameter errors of industrial robot based on Levenberg–Marquardt and Beetle Antennae Search algorithm. *Meas. Sci. Technol.* **2023**, *34*, 105024. [[CrossRef](#)]
16. Tian, A.Q.; Liu, F.F.; Lv, H.X. Snow Geese Algorithm: A novel migration-inspired meta-heuristic algorithm for constrained engineering optimization problems. *Appl. Math. Model.* **2024**, *126*, 327–347. [[CrossRef](#)]
17. Yan, J.; Pan, B.; Fu, Y. Ultrasound-guided prostate percutaneous intervention robot system and calibration by informative particle swarm optimization. *Front. Mech. Eng.* **2022**, *17*, 3. [[CrossRef](#)]
18. Zeng, C.D.; Qiu, Z.C.; Zhang, F.H.; Zhang, X.M. A novel error compensation method for a redundant parallel mechanism based on adaptive differential evolution algorithm and RBF neural network. *Precis. Eng.* **2025**, *94*, 26–42. [[CrossRef](#)]
19. Bian, H.; Li, C.; Liu, Y.; Tong, Y.; Bing, S.; Chen, J.; Ren, Q.; Zhang, Z. Improved snow geese algorithm for engineering applications and clustering optimization. *Sci. Rep.* **2025**, *15*, 4506. [[CrossRef](#)] [[PubMed](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.